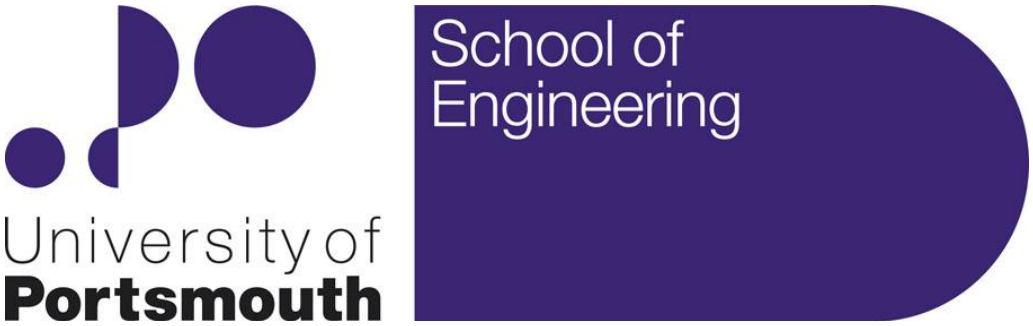


Location Based Guide Application For Windows Phone 7

Project Report

Jonas Koisti
up636310

Exchange Programme Eng
CEB3

2011/2012

Supervisor: Dr Linda Yang

Moderator: Dr Ioannis Kagalidis

School of Engineering
Faculty of Technology
May 2012

Abstract

Location based guide application for Windows Phone 7 is a final year computer engineering project that was done in the University of Portsmouth. This project consists of a report, logbook and the actual files used in the application. Project was done for the Windows Phone 7 operating system and will showcase the map implementation techniques used in the mobile phone platform. A lot of data is handled in this project, which consists of different venues around the city of Portsmouth. These venues are saved to a SQL compact edition database saved inside the mobile phone.

Different venues in this project can be browsed in different screens to help the users of this application to find venues appealing to them, in the city of Portsmouth. Application also helps people to find their way around the city by showing their position in the map, implemented using Bing maps service. Among the other venues, the University of Portsmouth buildings are also showcased in this application.

This report will talk about the design of this project, which was mainly done with the Expression Blend 4, and the different parts of the screen, how they work and why they were included in the project. All the programming was done by using C# language and the Visual Studio 2010 Express for Windows Phone 7. There is also a discussion part, where we'll talk about the project as a whole and what results were achieved and how they reflect to the original project plan. In the end of the the report, there are some good features that could be implemented in the future, if this project is worked on later.

Table of Contents

1. Introduction

There are currently three major mobile smartphone operating systems; Apple's IOS, Google's Android and Microsoft's Windows Phone 7 (WP7). From these three, the Windows Phone has the least amount of users and is by far the youngest having been around for less than two years. The amount of applications (App's) is rising rapidly and more and more developers are turning their sights towards the mobile platforms. With the release of the new Nokia phones running the WP7 operating system, the market share of WP7 phones is rising, but it has a long way to the numbers of iPhone and Android platform phones.

The amount of different types of applications you can do for mobile platforms is vast. You can roughly divide a mobile application in to two categories; games and applications that enhance the usage of the phone as a tool to react with ones surroundings. One type of application that is very useful and widely used amongst the application developers is a guide application using a map implementation that provides you with directions and places of interest in a specific location or a city.

Global map is a very common feature in today's mobile phone and is usually included as software on the phone. WP7 series has the Bing maps, which the user can use to get their own location, as well as a location of a restaurant or other places of interest. This project aims to create more comprehensive application which would give the user all the necessary information about the city of Portsmouth and places of interest with just a couple of clicks.

There are already a couple of other applications available, but they are done mainly for the big cities of the world. The best software that provides a map implementation is the Nokia Maps, which is installed by default on the new Nokia phones running WP7. Nokia Maps works well, but is not specifically aimed at any particular city. This leaves a market for the more specified applications and I think that the city of Portsmouth could use an application like this.

People carry their phones to almost everywhere with them, and that is the easiest place to find information when we don't have a computer at hand. Having an application that would show the restaurants, pubs, museums, cultural venues and the University campus in a couple of seconds is almost essential in these years that information is everything and has to be easily available. People want information and this application could be something that even the locals of Portsmouth could use daily.

2. Background Theory

The Windows Phone 7 operating system was introduced at the Mobile World Congress in Barcelona on February 15th 2010 and the first phones were released later that year on the 21st of October. WP7 uses a UI design called "Metro" that uses tiles to display apps and phone options. These tiles have the ability to display information and live updates on the front screen and tiles can also be made animated. The user can choose what tiles to see on the front page and arrange them to any order. Metro design is also going to be used in the new Windows 8 operating system, and was originally used in the Zune media software.

"Windows Phone 7 breaks the current smartphone convention to help people quickly and easily find and consume data, information and services from the Web and applications. The new phones are distinguished by unique design and integrated experiences built from Microsoft's deep portfolio such as Xbox LIVE, Microsoft Office Mobile, Zune, Windows Live, Bing and more."(Microsoft, 2010) Microsoft has really brought everything it has to offer for this phone and still after using it for over a year you can find new features you weren't previously aware of.

In this software we are using the Bing maps to show a map of the world and use the System.Device.Location namespace to connect and display the map. The namespaces used are all available from the WP7 SDK, which can be downloaded from the Microsoft web page. The SDK comes with two main software's; Blend and Visual Studio 2010 Express. Blend can be used to create and design WP7 applications using design tools and the .NET framework. Visual Studio provides you with templates to get a head start on your project and allows you to test your applications on the Windows Phone 7 emulator. Both Blend and Visual Studio work together, so you can open your Visual Studio project on Blend and any changes you make in Blend are automatically updated to your project on the Visual Studio. Visual Studio also enables you to work with the XNA framework which you can use to start creating your own mobile phone games.

2.1 Geo Coordinate Watcher

"The System.Device.Location namespace allows application developers to easily access the computer's location by using a single API. Location information may come from multiple providers, such as GPS, Wi-Fi triangulation, and cell phone tower triangulation. The System.Device.Location classes provide a single API to encapsulate the multiple location providers on a computer and support seamless prioritization and transitioning between them."(Microsoft, 2012)

Namespace System.Device.Location is used in two screens in this project; the location screen and add new venue screen. Geo coordinate watcher is initialized in both pages to get the users current location. Usage of this namespace requires a data connection or a Wi-Fi and using both gives the

best and most accurate result. High accuracy is used throughout the application and the movement of the phone is updated every ten meters.

2.2 Isolated Storage

Microsoft has introduced an isolated storage in the .NET framework to manage the data saved by the application. The data is always isolated from other applications by user and by assembly. The data saved can be of any type, and in this application a text file is saved that holds all the new locations added by the user.

Three main classes are provided to help you perform tasks that involve isolated storage:

- IsolatedStorageFile , which derives from IsolatedStorage, provides basic management of stored assembly and application files. An instance of the IsolatedStorageFile class represents a single store located in the file system.
- IsolatedStorageFileStream , which derives from System.IO.FileStream, provides access to the files in a store.
- IsolatedStorageScope is an enumeration that enables you to create and select a store with the appropriate isolation type.

(Microsoft, 2012)

IsolatedStorageFile is used to create the file used by the application and to open the file, when we want to read or write to it. IsolatedStorageFileStream handles the reading and writing with I/O stream.

3. Work Done and Results Achieved

“Think twice before developing your app.”(WindowsPhoneGeek, 2011) This is one of the main points in the article published 17th of January in 2011 discussing about what makes a WP7 application successful. Creating an application that no-one uses is a waste of time and money. Because there are so many applications out there, yours has to stand out from the masses by giving the user something that it wants to get back to again and again.

App, like any other software developed in the world, has to be practical and do what it’s supposed to be doing, and do that the best way it can. No reason to have a lot of different features in an application if none of them works to their potential and just does everything barely “ok”. Working with the mobile platform requires developers to take in to account the performance restrictions and always keep in mind the reliability and performance of the application.

Although the hardware in phones is improving by leaps and bounds, it is still far away from the Personal Computers (PC's) and laptops we use daily. Performance without reliability is nothing; even though reliability works a long way in the expense of performance, they should always be treated as equals. Windows Phone OS has a limit of ten seconds for starting an application. In that ten seconds, the app hast to start and be responsive for user input.

3.1 Design

User Interface (UI) is the first thing the user is going to see when opening the application. Everyone has heard sometime in their life about how important the first impression really is. Just like in real life, where the first impression can be made by the posture, clothing, handshake or the overall habitus of a person, in applications it's the UI that catches the eye of the user. When developing with WP7 it's good to remember to follow the Metro UI guidelines (Microsoft, 2012), that make it stand out from its competitors Apple's iPhone and Google's Android.

In this project, the main points when considering the design were;

- Appealing and clean –looking main page that opens when the application is started
- Simple and easy to use map screen
- The way that all the different venues are available for browsing, in this case, the list boxes

3.1.1 Main Page

This is the main page that opens when the application is started. In the middle you can see different categories to choose from, that include; Food & Drink, Museums & History, Theaters & Cinema and Shopping. There are also couple extra ones, to make the app a bit more informative. These include; Location, University of Portsmouth and About Portsmouth. There is also one feature that is hidden in the left bottom corner behind the letter A, which opens a screen, where you can add a new venue, that is not yet in the database, from your current location.

The background image is taken from the seaside of the Spinnaker Tower and is used throughout the whole application. The other pages use only the sky in the background and the Tower itself is removed from the picture

to make it easier to browse through the different lists in this application.

XAML code behind all the different buttons in the main page are; a rectangle with rounded corners and gradient brush on the background, text block with the different labels on top of that and the actual click event is handled by another rectangle on top of those two. The top most rectangles opacity is set to 0, so the user doesn't even see that it exists. The A on the left bottom corner is a button with content set to "A".

The main page is designed in Microsoft Expression Blend 4 and then opened in the Visual Studio to check and tweak all the margins and heights to make them exactly the same way and position them correctly. Blend is good software to design the applications, but in the end I think that Visual Studio works better with the actual XAML and C# code, although developer can also work on the code in Blend.

3.1.2 Location Page

Most of the location screen area is used by the map, as it is the most important thing in this page. The map loads to the middle of the screen as a road version of the whole world, using the Bing maps. Under the map there is a status text block that shows the current status of the location service, and gives a helping hint in the beginning to press the crosshair icon to start the location service.

At the beginning of the project, there were only two buttons at the bottom of the screen labeled "Start Tracking" and "Update Map". Towards the end of the project, the buttons were removed and an application bar was added to the bottom to enhance the Metro UI design factor and that decision also gave up more space for the actual map. In the application bar there are three different icons shown; a crosshair, refresh and settings –icon. By clicking on the three points on the right, the application bar expands a little to show the labels of those buttons. To someone's eye the application bar may seem hard to figure out, but it is widely used in the WP7 platform and you see it in almost all applications. Using easy to understand icons enhances the usability and performance.

The crosshair –like icon has the label "find me!" and while it is meant to show the user where he/she is, it also starts the actual location service in the background that enables the second button to work. The refresh button gets all the different locations from the phones database and shows them on the map with color coded pushpins. The application bars last button which uses the well-known settings –icon is there to change the way the map looks like. The default way that the map is shown is basic road style with labels, but when that button is pressed, the map changes to an aerial view and then vice versa.

3.1.3 Browsing the Venues

As the user starts the application and sees the main page, a decision has to be made on what type of places to look for. All the different lists of venues use the same kind of design to enhance the coherence of the application. As an example let's look at the food & drink screen, which uses the pivot control and a list box. When the amount of venues increases, the list box automatically adds a scroll bar which can be seen while scrolling up or down the list with a finger.

All the different types of venues are color coded and that color can be found next to the names of venues under a crosshair icon. The icon is added for an extra feature to find the venue from map quickly without actually checking the information page. Text in this page is white on the blue background which makes it stand out and is easy to read. One of the main points was to make the page look clean and simple, but with enough information to help the user choose a venue before even looking at the information page for the full description, which is why the short description seen in these pages has to be short and informative.

3.2 Implementation

In this software, there are eleven different screens that can be seen when using the application. Each of those screens consists of a XAML and C# file. The XAML file provides the frontend of the screen, and can be seen in the Visual Studio showing an image of what it would look like in the actual phone. This image changes in real-time which make small UI design changes fast and easy with the XAML. The C# file handles the backend and everything that is happening in the background. The UI can also be done with the C#, but is a lot more difficult as the developer then doesn't have a reference image to look and check the results.

Portsmouth Guide application does all the UI design in the XAML file throughout the whole software and the C# is used to transfer information from screen to screen and handle the data queries from the SQL database. Lot of the information shown on the screen, like name of a restaurant, is bound to a variable *name* that is then queried and attached to that variable in the C# code.

All of the libraries used in this project are included in the Visual Studio 2010 Express for Windows Phone SDK, which is used to develop this application. Parts of the software like location, database and image required libraries that weren't used by default and those had to be referenced when needed.

XAML page in this project consists of a layout root where all the page content is placed and has a background image of the screen. Inside this layout root there are two panels; Title panel and Content panel. Title panel contains a text block showing the title of the screen, using Cambria – font and static style of title text. Content panel is the panel that has all the different information that is included in the XAML code. This content is usually made up from list boxes, text blocks, images or map implementations. In the next chapters we'll go through the different screens that are used in this software and how they work and communicate with each other.

3.2.1 App Class

App is the base class of the application and handles the running of the application. When a new project is created, this class is automatically created to connect to the root frame of the phone application and is responsible for the starting and closing of the application. The connection to the SQL database is made in the constructor. The existence of the database is checked and if the software is ran for the first time, the database is created and all the data is added. The venues in this project are added one by one, which in the first time makes the starting of the application longer, but when the application is ran for the second time, if just checks that database already exists and no new data need to be added. After the database is created, we try to create a new AppViewModel called viewModel with the connection string to the database.

3.2.2 View Model

App view model constructs all the observable collections used by the application and loads the data from database to these collections. When this class' method LoadCollectionsFromDatabase() is called from the App class, it starts going through the database.

First thing the method does, is all the type Venues from the database and then loads them to their respective collections depending on their type. Some collections have to be united, for example to make all the food and drink places appear on the same list, because Silverlight doesn't support multi binding from many lists or collections.

3.2.3 Main Page

As being the first thing the user sees, this page has to be looking good and appealing. It definitely is the best looking of the screens but at the same time it's the simplest one also. Background in this page is the Spinnaker Tower, photo that is used in all the other pages as well but without the Tower itself, which gives the application a nice sunny and contiguous feel. Content panel, like talked in the design section, consists of the different rectangles and text blocks. What really makes this page work, are the Event triggers in the invisible rectangles.

```
<Rectangle x:Name="restClickRect" Height="75" Margin="30,176,30,0" Stroke="Black"
StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
        <i:Interaction.Triggers>
            <i:EventTrigger EventName="MouseLeftButtonDown">
                <ec:NavigateToPageAction TargetPage="/Screens/RestaurantsScreen.xaml"/>
            </i:EventTrigger>
        </i:Interaction.Triggers>
 </Rectangle>
```

The rectangle here includes an EventTrigger that waits for an event named MouseLeftButtonDown, which in this case means that it does something when the user taps inside this rectangle with a finger. That something is a NavigateToPageAction that navigates to a screen, which URI is in the TargetPage variable, this case it's the Restaurant screen.

3.2.4 Location Page

One of the biggest challenges of the location screen was that it was accessible from so many different screens. Have to be taken in to account, that many different types of information would be handled by the screen. In the constructor of the location page, the GeoCoordinateWatcher, which is used to handle the location data, is initialized and movement threshold set to ten meters. Tracking is set to false in the beginning and two event handlers are created for the watchers position and status changes.

As the watchers status changes, the text on the screen under the map is changed to show the user the current status. There are four different states for the watchers status and if the status is disabled, it also checks if the user has disabled the service, or if it doesn't work on the device. WatcherPositionChanged method checks if the tracking is on and then gets the users current location and adds a pushpin to the map to show the users current location. Location service is started with a thread that is given thirty seconds to start.

When the user clicks the update button on the application bar, updateMe_click eventhandler is called. If tracking is on, all the venues are loaded from the database and added to the pushpin collections. When all the venues are successfully loaded, the pushpins are layed out on the map in pushpin layer and the view rectangle of the map set to zoom on those pushpins instead of showing the whole world on the map.

Override method OnNavigatedTo is ran every time when the user navigates to the location page but is there in this project to handle the incoming data from other screens. Data is transferred to this page from the information screen and the venue browsing pages. OnNavigatedTo checks if there is latitude coordinates in the incoming stream from the navigation service and if yes, it checks the database and finds which data is in question, that has these coordinates.

The different color coded pushpins are chosen by checking the type of the venue and setting the correct pushpin. This is all done in a switch case statement, which also has a default of message box appearing if the type was unrecognized.

3.2.5 Pivot Controller Pages

Three screens in the application are using the Pivot control which enables user to slide finger on the screen left or right to change to another page. These are the University, Restaurants and Shopping –screens. Pivot is one of the great Metro UI design control templates that enable the developer to show large amounts of information on one screen and at the same time, can make it easily available to the user. It prevents the use of a very long list box, as the screens information can be divided in to sections.

3.2.5.1 University Page

The University Screen has three pivot items labelled; info, history and buildings. Info and history consist of list box showing information and history about the University of Portsmouth, which is copied from the University's homepage. The text in the list box is wrapped, the line height is increased and a margin of 8px is also added to make it easier to read.

On the XAML side, the last pivot item buildings has its own template called "BuildingsListBoxItemTemplate" that is created as a resource for this page. This template consists of three rows and two columns. In the first column spanning the whole three rows is the name of the building bound from UniversityBuildings –observable collection. Just in the first row of the second column is a crosshair icon, which will open the locations page showing the information about the users current location and the location of the building in question. The item template is set as a static resource to the list box inside the buildings pivot item.

In the C# code, there are two tap gesture events, that check for the users input. If the user touches the name of the building, the application then check the name tapped and while navigating to the information page, it also sends the name of the building, so the information page then knows which building was selected.

If the crosshair icon is pressed, the latitude and longitude of that selected building is sent to the location page, again with the NavigationService.

3.2.5.2 Restaurants Page

Using the pivot control in the restaurant screen is a good way to categorize the different food and drink places around Portsmouth. In this application the different categories are; restaurants, pubs, coffee and all. All of these four items use the same item template, but the values inside are just bound to different collections in the database.

One item consists of three rows and three columns, inside which are an image and two text blocks. Image is situated in the first row and the first column on the left top corner with height and width of 50. Next to the image on the second column is a text block which contains the name of the venue, bound to a variable "name". Under the image and the name is another text block that contains a short description which is bound to a variable "description_short". For each of these four pivot items, a list box is created in which the ItemsSource is then bound to their respective observable collections in the database.

The C# handles the event of selecting a venue from the list. If a venue is chosen from any of the pivot item list boxes, a NavigationService is called to navigate to the information screen, with the name and type of the restaurant also sent.

3.2.5.3 Shopping Page

The layout of the shopping screen is the same as the restaurant screen, except that it only has three pivot items labelled as; fashion, sport and all. These items all have a list box which uses a template, like in the restaurant screen. Again the data in the list box is bound their respective observable collections created from the database. When any of the items is chosen, the name and type are again transferred with the NavigationService.

3.2.6 List Pages

List screens are a bit simpler in style, compared to the pivot controlled screen, as they only have one list in the screen consisting of the different venues under that type. When browsing through theatres and historical attractions, the amount of these places in the city of Portsmouth is limited to only a handful, which allows us to show them all in one list box. Theatres & Cinema and Museums & History –screens are using the same item template already used in the other pivot control screens and their list boxes. The name and short description are bound to their own collections of venues labelled as “Theatre” and “History”.

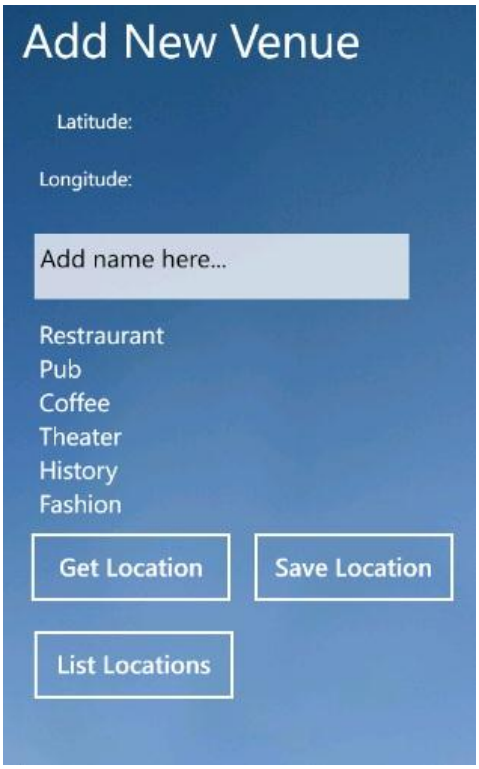
```
private void HistoryListBox_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

    var dataOfPlace = HistoryListBox.SelectedItem as Venues;

    NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),

        UriKind.RelativeOrAbsolute));

}
```

The method here checks which venue was selected and then sends the name and the type to the information screen with the NavigationService. This part of code is widely used in the application to transfer data between screens.

3.2.7 Add Venue Page

In the main page there is a capital letter a in the left bottom corner of the screen, purposely hidden. This button navigates to a screen where you can add your current location as a venue to a list in the data storage of the phone. It was constructed to make adding new venues easier for the developer.

The AddVenueScreen is unlike the other screens, its design is simple and basic but it still provides developer the needed features for adding a new venue. The page is made to be informative and easy to use, which shows from the buttons and the text block in the middle.



Under the title, there are two text blocks, latitude and longitude which show the user the current location when the button “Get Location” is pressed. After the user has the coordinates, the name can be added and type of place selected. When all this information is added and selected the user can press “Save Location”. All the fields will go back to the stage we can see on the right after the saving is completed. This gives the user the option to add another place fast and easy. “List Locations” button in the bottom of the page navigates the user to a new screen, where all the new added venues are shown.

When this screen is opened and initialized, the software tries to create a new isolated storage file after checking if it already exists. If the file already exists, we write all the venues already in the file, to our list “ListOfLocations”. If for some reason, the isolated storage doesn’t work, we get an error message “Isolated Storage not working!” and the user can try again later. When the initialization is completed, the app is ready to take user input.

To get the users current location, a new Geo coordinate watcher is started whit accuracy high (takes longer to find the correct position of the user, but is more accurate that low). Status and position event handlers are also initialized to keep track of the user if he’s moving. If the position of the phone changes, the latitude and longitude are updated. Value of the movement threshold for position changed is ten meters.

Pressing the save location, checks that a type is chosen for the venue and then saves this venue to the list of locations and to the isolated storage as a new venue with name, type and coordinates. Every time a new venue is added the whole text file is updated with the list of venues. This list can be viewed by pressing the third button, which navigates to another screen.

ShowNewListScreen connects to the isolated storage, looks for “locations.txt” and then while reading the file line by line, adds each line to our list with members as strings. This screen has a list box with a text block and all the values read from file are shown in the list box for the user or developer to see.

4. Discussion

What has to be thought about when discussing the outcome of this project is that it’s as big as you make it to be. There are numerous possible implementations to add and just when you think you’ve done all you wanted; you come up with another idea. The different aspects of this project give a good basis for developing ones programming skills while at the same time making it fun while doing it. It all starts with the design and the overall project plan. Due to the short amount of time, developer really has to choose on what to concentrate on. The fact is, not everything can be done and this type of project is never perfect, it just keeps growing until we run out of ideas.

One of the downsides of working with Windows Phone 7 is that, although developer can use the emulator to test the software, one has to be a registered Windows Phone developer and this service costs you 75€ a year. The emulator itself is really good and the Mango update even brought tilt testing capabilities which is useful when testing applications or games that use the Accelerometer. Using the actual phone for testing is just too much fun compared to anything else, it takes debugging and testing the software to a whole another level. One of the greatest things is that you can actually test your project anywhere.

There are requirements for developing for the phone and that usually leads to very extensive testing, which is very time consuming. Testing has to be done at short intervals and you have to be aware of everything that is happening. Many times during this project, came a time when nothing seemed to go forward however long they were worked on. The Windows Phone 7 is such a young platform, it doesn’t have that wide of a developer community as the older programming languages. Many parts of the application take a long time to understand and there are times, when even Google doesn’t help when there’s an unidentified error on the screen. Visual Studio is also very effective on spotting errors in the code and these errors prevent the application from launching. With a generic error message and no running software to debug, finding the actual error can take a long time and a lot of searching.

4.1 Results Achieved

Going in to this project, there were a lot of ideas running around my head, but sadly I wasn’t able to implement all that I wanted. Too many times I faced an obstacle that took too long to overcome. The results I achieved are good in my opinion, but could have been better, with some

extra time and know-how. I am fairly pleased on what I achieved and will be working on this on the future to make the application even better. While many of the features and the overall quality may seem basic and quite simple, the project was not easy to do for a developer that didn't know anything about mobile phone application development before starting the project. The location screen was fairly easy to implement but as more features were added to the screen, the code kept expanding and many times the old code had to be changed in order for the new features to work.

Modifying old code brings the problem of changing small things in the code that actually affect to a lot of different methods and that way, make the process longer than it has to be. While adding the color coded pushpins, the whole way of loading the data on the location page had to be changed. However much research has been done before starting the project there are always going to be surprises and problems.

In the end the design looks good and there is a warm feeling about the first page and especially the Spinnaker Tower. The design around the application works well, navigation is fluent and finding information is easy. Launching the application is fast and it's evident that performance was taken in to account in every step of the way on developing this project.

5. Future Work

Guide applications on mobile phones are still quite young in comparison to a GPS used in cars and have the potential of being something bigger with lots of different information. This software is lacking in some features that the big companies have in their map applications, but at the same time, the locality of it stands out. The biggest shortcoming of this application is the local database it's using.

As a future work the first things to do would be to implement a network database with a frontend that the administrator or even the everyday user could use to add new venues and remove old ones by just using their web browser. A whole review website could be written around this program, although it usually goes the other way around as these rating websites are very common and have been around for years now.

In this program like in the city of Portsmouth itself, the university plays a big part. One of the groups that this application is aimed for are the new students coming to the University of Portsmouth, in need of help in their first few months here finding their way around. Already implemented in this application is the ability to locate different university buildings, but another great thing would be to check your timetable straight from the application. This could be achieved by implementing a log in page on the application, which would then connect to the portal and query the timetable, straight to the phone. This can already be achieved with phone by starting a web browser, but this way would be faster and easier.

6. Bibliography

[1] IGN.2012.Windows Phone 7 Wiki Guide.Retrieved May 23rd, 2012, from http://uk.ign.com/wikis/windows-phone

7. References

[1] Microsoft.2012. System Device Location Namespace. Retrieved May 23rd, 2012, from http://msdn.microsoft.com/en-us/library/ee426011

[2] Microsoft.2012. Isolated Storage. Retrieved May 23rd, 2012, from http://msdn.microsoft.com/en-us/library/3ak841sy%28v=vs.100%29.aspx

[3] WindowsPhoneGeek.2011. What makes a WP7 App successful. Retrieved May 23st, 2012, from http://www.windowsphonegeek.com/articles/What-makes-a-WP7-App-successful

[4] Microsoft.2012. User Experience Design Guidelines for Windows Phone.Retrieved May 24th,2012, from http://msdn.microsoft.com/en-us/library/hh202915%28v=VS.92%29.aspx

8. Appendices

Appendix I: Source Code

App.xaml

```
<Application
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
             xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
             x:Class="Portsmouth_Guide.App">

             <!--Application Resources-->
             <Application.Resources>
             </Application.Resources>

             <Application.ApplicationLifetimeObjects>
                          <!--Required object that handles lifetime events for the
application-->
                          <shell:PhoneApplicationService
```

```
                                    Launching="Application_Launching"
Closing="Application_Closing"
                                    Activated="Application_Activated"
Deactivated="Application_Deactivated"/>
            </Application.ApplicationLifetimeObjects>
</Application>
```

App.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Navigation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Shell;
using Portsmouth_Guide.Model;
using Portsmouth_Guide.ViewModel;

namespace Portsmouth_Guide
{
    public partial class App : Application
    {
        /// <summary>
        /// Provides easy access to the root frame of the Phone Application.
        /// </summary>
        /// <returns>The root frame of the Phone Application.</returns>
        public PhoneApplicationFrame RootFrame { get; private set; }

        private static AppViewModel viewModel;
        public static AppViewModel ViewModel
        {
            get { return viewModel; }
        }

        /// <summary>
        /// Constructor for the Application object.
        /// </summary>
        public App()
        {
            // Global handler for uncaught exceptions.
            // Note that exceptions thrown by ApplicationBarItem.Click will not get caught
here.
            UnhandledException += Application_UnhandledException;

            // Show graphics profiling information while debugging.
            //if (System.Diagnostics.Debugger.IsAttached)
            //{
            //    // Display the current frame rate counters.
            //    Application.Current.Host.Settings.EnableFrameRateCounter = true;

            //    // Show the areas of the app that are being redrawn in each frame.
            //    //Application.Current.Host.Settings.EnableRedrawRegions = true;
```

```
            //    // Enable non-production analysis visualization mode,
            //    // which shows areas of a page that are being GPU accelerated with a
colored overlay.
            //    //Application.Current.Host.Settings.EnableCacheVisualization = true;
            //}

            // Standard Silverlight initialization
            InitializeComponent();

            // Phone-specific initialization
            InitializePhoneApplication();

            // Database connection string
            string DBConnectionString = "Data Source=isostore:/appData.sdf";

            // Try to connect to the database and check if it exists
            using (AppDataContext db = new AppDataContext(DBConnectionString))
            {
                if (!db.DatabaseExists())
                {
                    // If the database didn't exist, create it and add all the Venues
                    db.CreateDatabase();

                    var restaurant1 = new Venues { name = "Raj Dulal", t = "Restaurant",
imageUri = "", description_short = "Indian takeaway restaurant", description = "Raj Dulal in
Southsea, Portsmouth is an authentic Indian Takeaway serving the finest in home style Indian
Cuisine.     We provide freshly cooked food for you to enjoy in the comfort of your own
homes.   Order Online for FREE Delivery or Collection and earn loyalty points.", webaddress
= "http://www.rajdulal.co.uk", latitude = 50.7932242722713, longitude = -1.07659565459686 };

                    var restaurant2 = new Venues { name = "La Tasca", t = "Restaurant",
imageUri = "", description_short = "Fantastic Spanish restaurant", description = "This is a
sample what the text could be like and here you would put all the information you want the
user to see about the restaurant..", webaddress = "", latitude = 50.792808, longitude = -
1.106867 };

                    var pub1 = new Venues { name = "The Duke Of Devonshire", t = "Pub",
imageUri = "", description_short = "Old style pub straight from the 70's", description =
"From outside, this pub looks small and very old. The looks don't fool, as you walk in and
find yourself walking back tens of years. This is pub is very popular with the locals.",
webaddress = "", latitude = 50.78670681, longitude = -1.07780725 };

                    var coffee1 = new Venues { name = "Coco Cafe", t = "Coffee", imageUri =
"", description_short = "University area coffee shop with proper filter coffee", description
= "Coffee shop with a nice atmosphere and lots of tables. Good food for lunch and lots of
students go there to work on their schoolwork", webaddress = "", latitude =
50.7946938509595, longitude = -1.09628604725105};

                    var history1 = new Venues { name = "National History Museum", t =
"History", imageUri = "", description_short = "Amazing place with lots to see", description
= "Description of this historic attraction goes here and it says something that would make
the tourist to go there.", webaddress = "", latitude = 50.79469385034321, longitude = -
1.09628604725105 };

                    var shop1 = new Venues { name = "Fashion Clothing", t = "Fashion",
imageUri = "", description_short = "Quality clothing for everyone", description = "Here goes
some description of the actual company that sells the clothes. What kind of clothes and to
whom the clothes are made for.", webaddress = "www.microsoft.com", latitude = 50.794454,
longitude = -1.104234 };

                    var shop2 = new Venues { name = "Sports Equipment", t = "Sport",
imageUri = "", description_short = "The largest tennis racket selection in the country!",
```

```
description = "Here we have the biggest tennis racket selection of the country with some great prices and all year offers.", webaddress = "www.microsoft.com", latitude = 50.794454, longitude = -1.104172 };

                    var theatre1 = new Venues { name = "Vue Cinema", t = "Theatre", imageUri = "", description_short = "The biggest cinema in the city", description = "A big new cinema located in the Gunwharf Quays shopping area. Lots of screen and all the new movies.", webaddress = "www.microsoft.com", latitude = 50.780473, longitude = -1.104172 };

                    var uni1 = new Venues {name = "Anglesea Building", t = "University", imageUri = "/images/university/anglesea.jpg", description_short = "", description = "University of Portsmouth\nAnglesea Road\nPortsmouth\nPO1 3DJ", webaddress = "", latitude = 50.7946938503425, longitude = -1.09628604725105};

                    var uni2 = new Venues {name = "Buckingham Building", t = "University", imageUri = "/images/university/buckingham.jpg", description_short = "", description = "University of Portsmouth\nLion Terrace\nPortsmouth\nPO1 3HE", webaddress = "", latitude = 50.7946938503426, longitude = -1.09628604725105};

                    var uni3 = new Venues { name = "Burnaby Building", t = "University", imageUri = "/images/university/burnaby.jpg", description_short = "", description = "University of Portsmouth\nBurnaby Road\nPortsmouth\nPO1 3QL", webaddress = "", latitude = 50.7946938503427, longitude = -1.09628604725105 };

                    var uni4 = new Venues { name = "Dennis Sciama Building", t = "University", imageUri = "/images/university/dennis_sciama.jpg", description_short = "", description = "University of Portsmouth\nBurnaby Road\nPortsmouth\nPO1 3FX", webaddress = "", latitude = 50.7946938503428, longitude = -1.09628604725105 };

                    var uni5 = new Venues { name = "Eldon Building", t = "University", imageUri = "/images/university/eldon.jpg", description_short = "", description = "University of Portsmouth\nWinston Churchill Avenue\nPortsmouth\nPO1 2DJ", webaddress = "", latitude = 50.7946938503429, longitude = -1.09628604725105 };

                    var uni6 = new Venues { name = "King Henry Building", t = "University", imageUri = "/images/university/king_henry.jpg", description_short = "", description = "University of Portsmouth\nKing Henry I Street\nPortsmouth\nPO1 2DY", webaddress = "", latitude = 50.7946938503430, longitude = -1.09628604725105 };
                    var uni7 = new Venues { name = "Institute of Marine Sciences at Langstone Harbour", t = "University", imageUri = "/images/university/marine_labs.jpg", description_short = "", description = "University of Portsmouth\nFerry Road\nEastney\nHampshire\nPO4 9LY", webaddress = "", latitude = 50.7946938503431, longitude = -1.09628604725105 };

                    var uni8 = new Venues { name = "Lion Gate Building", t = "University", imageUri = "/images/university/lion_gate.jpg", description_short = "", description = "University of Portsmouth\nLion Terrace\nPortsmouth\nPO1 3HF", webaddress = "", latitude = 50.7946938503432, longitude = -1.09628604725105 };

                    var uni9 = new Venues { name = "Nuffield Centre", t = "University", imageUri = "/images/university/nuffield_centre.jpg", description_short = "", description = "University of Portsmouth\nSt Michael's Road\nPortsmouth\nPO1 2ED", webaddress = "", latitude = 50.7946938503433, longitude = -1.09628604725105 };

                    var uni10 = new Venues { name = "Park Building", t = "University", imageUri = "/images/university/park.jpg", description_short = "", description = "University of Portsmouth\nKing Henry 1 Street\nPortsmouth\nPO1 2DZ", webaddress = "", latitude = 50.7946938503434, longitude = -1.09628604725105 };

                    var uni11 = new Venues { name = "Portland Building", t = "University", imageUri = "/images/university/portland.jpg", description_short = "", description =
```

```
"University of Portsmouth\nPortland Street\nPortsmouth\nPO1 3AH", webaddress = "", latitude
= 50.7946938503435, longitude = -1.09628604725105 };

                    var uni12 = new Venues { name = "Richmond Building", t = "University",
imageUri = "/images/university/richmond.jpg", description_short = "", description =
"University of Portsmouth\nPortland Street\nPortsmouth\nPO1 3DE", webaddress = "", latitude
= 50.7946938503436, longitude = -1.09628604725105 };

                    var uni13 = new Venues { name = "Spinnaker Building", t = "University",
imageUri = "/images/university/spinnaker.jpg", description_short = "", description =
"University of Portsmouth\nCambridge Road\nPortsmouth\nPO1 2ER", webaddress = "", latitude =
50.7946938503437, longitude = -1.09628604725105 };

                    var uni14 = new Venues { name = "St George's Building", t =
"University", imageUri = "/images/university/st_george.jpg", description_short = "",
description = "University of Portsmouth\n141 High Street\nPortsmouth\nPO1 2HY", webaddress =
"", latitude = 50.7946938503438, longitude = -1.09628604725105 };

                    var uni15 = new Venues { name = "St Michael's Building", t =
"University", imageUri = "/images/university/st_michael.jpg", description_short = "",
description = "University of Portsmouth\nWhite Swan Road\nPortsmouth\nPO1 2DT", webaddress =
"", latitude = 50.7946938503439, longitude = -1.09628604725105 };

                    var uni16 = new Venues { name = "St Paul's Sports Centre", t =
"University", imageUri = "/images/university/st_paul.jpg", description_short = "",
description = "University of Portsmouth\nSt Paul's Road\nSouthsea\nPO5 4AQ", webaddress =
"", latitude = 50.7946938503440, longitude = -1.09628604725105 };

                    var uni17 = new Venues { name = "Student Centre", t = "University",
imageUri = "/images/university/student_centre.jpg", description_short = "", description =
"University of Portsmouth\nCambridge Road\nPortsmouth\nPO1 2EF", webaddress = "", latitude =
50.7946938503441, longitude = -1.09628604725105 };

                    var uni18 = new Venues { name = "University Library", t = "University",
imageUri = "/images/university/library.jpg", description_short = "", description =
"University of Portsmouth\nCambridge Road\nPortsmouth\nPO1 2ST", webaddress = "", latitude =
50.7946938503442, longitude = -1.09628604725105 };

                    var uni19 = new Venues { name = "William Beatty Building", t =
"University", imageUri = "/images/university/william_beatty.jpg", description_short = "",
description = "University of Portsmouth\nHampshire Terrace\nPortsmouth\nPO1 2QG", webaddress
= "", latitude = 50.7946938503443, longitude = -1.09628604725105 };


                    // Adding the Venues type variables to the database
                    db.Venues.InsertOnSubmit(restaurant1);
                    db.Venues.InsertOnSubmit(restaurant2);
                    db.Venues.InsertOnSubmit(pub1);
                    db.Venues.InsertOnSubmit(coffee1);
                    db.Venues.InsertOnSubmit(history1);
                    db.Venues.InsertOnSubmit(shop1);
                    db.Venues.InsertOnSubmit(shop2);
                    db.Venues.InsertOnSubmit(theatre1);

                    db.Venues.InsertOnSubmit(uni1);
                    db.Venues.InsertOnSubmit(uni2);
                    db.Venues.InsertOnSubmit(uni3);
                    db.Venues.InsertOnSubmit(uni4);
                    db.Venues.InsertOnSubmit(uni5);
                    db.Venues.InsertOnSubmit(uni6);
                    db.Venues.InsertOnSubmit(uni7);
                    db.Venues.InsertOnSubmit(uni8);
```

```
                db.Venues.InsertOnSubmit(uni9);
                db.Venues.InsertOnSubmit(uni10);
                db.Venues.InsertOnSubmit(uni11);
                db.Venues.InsertOnSubmit(uni12);
                db.Venues.InsertOnSubmit(uni13);
                db.Venues.InsertOnSubmit(uni14);
                db.Venues.InsertOnSubmit(uni15);
                db.Venues.InsertOnSubmit(uni16);
                db.Venues.InsertOnSubmit(uni17);
                db.Venues.InsertOnSubmit(uni18);
                db.Venues.InsertOnSubmit(uni19);

                // Confirm the changes to the database
                db.SubmitChanges();

            }
        }

        // Load the collection from database by calling the LoadCollectionsFromDatabase
method on the AppViewModel class
        viewModel = new AppViewModel(DBConnectionString);
        viewModel.LoadCollectionsFromDatabase();

    }

    // Code to execute when the application is launching (eg, from Start)
    // This code will not execute when the application is reactivated
    private void Application_Launching(object sender, LaunchingEventArgs e)
    {
    }

    // Code to execute when the application is activated (brought to foreground)
    // This code will not execute when the application is first launched
    private void Application_Activated(object sender, ActivatedEventArgs e)
    {
    }

    // Code to execute when the application is deactivated (sent to background)
    // This code will not execute when the application is closing
    private void Application_Deactivated(object sender, DeactivatedEventArgs e)
    {
    }

    // Code to execute when the application is closing (eg, user hit Back)
    // This code will not execute when the application is deactivated
    private void Application_Closing(object sender, ClosingEventArgs e)
    {
    }

    // Code to execute if a navigation fails
    private void RootFrame_NavigationFailed(object sender, NavigationFailedEventArgs e)
    {
        if (System.Diagnostics.Debugger.IsAttached)
        {
            // A navigation has failed; break into the debugger
            System.Diagnostics.Debugger.Break();
        }
    }

    // Code to execute on Unhandled Exceptions
    private void Application_UnhandledException(object sender,
ApplicationUnhandledExceptionEventArgs e)
```

```
        {
            if (System.Diagnostics.Debugger.IsAttached)
            {
                // An unhandled exception has occurred; break into the debugger
                System.Diagnostics.Debugger.Break();
            }
        }

        #region Phone application initialization

        // Avoid double-initialization
        private bool phoneApplicationInitialized = false;

        // Do not add any additional code to this method
        private void InitializePhoneApplication()
        {
            if (phoneApplicationInitialized)
                return;

            // Create the frame but don't set it as RootVisual yet; this allows the splash
            // screen to remain active until the application is ready to render.
            RootFrame = new PhoneApplicationFrame();
            RootFrame.Navigated += CompleteInitializePhoneApplication;

            // Handle navigation failures
            RootFrame.NavigationFailed += RootFrame_NavigationFailed;

            // Ensure we don't initialize again
            phoneApplicationInitialized = true;
        }

        // Do not add any additional code to this method
        private void CompleteInitializePhoneApplication(object sender, NavigationEventArgs
e)
        {
            // Set the root visual to allow the application to render
            if (RootVisual != RootFrame)
                RootVisual = RootFrame;

            // Remove this handler since it is no longer needed
            RootFrame.Navigated -= CompleteInitializePhoneApplication;
        }

        #endregion
    }
}
```

AppViewModel.cs

```
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Linq;
using Portsmouth_Guide.Model;

namespace Portsmouth_Guide.ViewModel
{
```

```
    // In this class the software first creates a private instance of the AppDataContext
appDB
    // and the a public class AppViewModel, that it uses to connect to that private
DataContext
    // Instance of private ObservableCollection<Restaurant> is created and the getters and
setters for that.
    public class AppViewModel
    {
        private AppDataContext appDB;

        // Method to make the database queries
        public AppViewModel(string appDBConnectionString)
        {
            appDB = new AppDataContext(appDBConnectionString);
        }

        // Here we create the data collections for the different venues with the type of
Venues -class
        private ObservableCollection<Venues> _allVenues;
        public ObservableCollection<Venues> AllVenues
        {
            get { return _allVenues; }
            set { _allVenues = value; }
        }

        private ObservableCollection<Venues> _restaurantPlaces;
        public ObservableCollection<Venues> RestaurantPlaces
        {
            get { return _restaurantPlaces; }
            set { _restaurantPlaces = value; }
        }

        private ObservableCollection<Venues> _pubPlaces;
        public ObservableCollection<Venues> PubPlaces
        {
            get { return _pubPlaces; }
            set { _pubPlaces = value; }
        }

        private ObservableCollection<Venues> _coffeePlaces;
        public ObservableCollection<Venues> CoffeePlaces
        {
            get { return _coffeePlaces; }
            set { _coffeePlaces = value; }
        }

        private ObservableCollection<Venues> _allRestaurantPlaces;
        public ObservableCollection<Venues> AllRestaurantPlaces
        {
            get { return _allRestaurantPlaces; }
            set { _allRestaurantPlaces = value; }
        }

        private ObservableCollection<Venues> _theatrePlaces;
        public ObservableCollection<Venues> TheatrePlaces
        {
            get { return _theatrePlaces; }
            set { _theatrePlaces = value; }
        }

        private ObservableCollection<Venues> _historyPlaces;
        public ObservableCollection<Venues> HistoryPlaces
```

```
        {
            get { return _historyPlaces; }
            set { _historyPlaces = value; }
        }

        private ObservableCollection<Venues> _allShopPlaces;
        public ObservableCollection<Venues> AllShopPlaces
        {
            get { return _allShopPlaces; }
            set { _allShopPlaces = value; }
        }

        private ObservableCollection<Venues> _fashionShops;
        public ObservableCollection<Venues> FashionShops
        {
            get { return _fashionShops; }
            set { _fashionShops = value; }
        }

        private ObservableCollection<Venues> _sportsShops;
        public ObservableCollection<Venues> SportsShops
        {
            get { return _sportsShops; }
            set { _sportsShops = value; }
        }

        private ObservableCollection<Venues> _universityBuildings;
        public ObservableCollection<Venues> UniversityBuildings
        {
            get { return _universityBuildings; }
            set { _universityBuildings = value; }
        }

        private ObservableCollection<CustomPushPin> _customPushPins;
        public ObservableCollection<CustomPushPin> CustomPushPins
        {
            get { return _customPushPins; }
            set { _customPushPins = value; }
        }

        // Public class to load the collections from database using LinQ
        // After the database is queried, save it to the ObservableCollection
        public void LoadCollectionsFromDatabase()
        {
            var venuesInDB = from Venues ven in appDB.Venues
                             select ven;

            var restaurantsInDb = from Venues r in appDB.Venues where r.t == "Restaurant"
select r;
            var pubsInDb = from Venues p in appDB.Venues where p.t == "Pub" select p;
            var coffeeShopsInDb = from Venues c in appDB.Venues where c.t == "Coffee" select
c;
            var fashionShopsInDb = from Venues f in appDB.Venues where f.t == "Fashion"
select f;
            var sportShopsInDb = from Venues s in appDB.Venues where s.t == "Sport" select
s;
            var historicInDb = from Venues h in appDB.Venues where h.t == "History" select
h;
            var theatresInDb = from Venues t in appDB.Venues where t.t == "Theatre" select
t;
            var universityBuildingsInDb = from Venues u in appDB.Venues where u.t ==
"University" select u;
```

```
            AllVenues = new ObservableCollection<Venues>(venuesInDB);

            RestaurantPlaces = new ObservableCollection<Venues>(restaurantsInDb);
            PubPlaces = new ObservableCollection<Venues>(pubsInDb);
            CoffeePlaces = new ObservableCollection<Venues>(coffeeShopsInDb);
            AllRestaurantPlaces = new
ObservableCollection<Venues>(restaurantsInDb.Union(pubsInDb));
            AllRestaurantPlaces = new
ObservableCollection<Venues>(AllRestaurantPlaces.Union(coffeeShopsInDb));
            FashionShops = new ObservableCollection<Venues>(fashionShopsInDb);
            SportsShops = new ObservableCollection<Venues>(sportShopsInDb);
            AllShopPlaces = new
ObservableCollection<Venues>(fashionShopsInDb.Union(sportShopsInDb));
            HistoryPlaces = new ObservableCollection<Venues>(historicInDb);
            TheatrePlaces = new ObservableCollection<Venues>(theatresInDb);
            UniversityBuildings = new ObservableCollection<Venues>(universityBuildingsInDb);
        }

    }
}
```

AppDataContext.cs

```
using System;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Ink;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using System.ComponentModel;

namespace Portsmouth_Guide.Model
{
    public class AppDataContext : DataContext
    {
        public AppDataContext(string ConnectionString) : base(ConnectionString)
        { }

        public Table<Venues> Venues;
    }
}
```

Venues.cs

```
using System;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Ink;
using System.Windows.Input;
using System.Windows.Media;
```

```
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Data.Linq;
using System.Data.Linq.Mapping;

namespace Portsmouth_Guide
{
    [Table]
    public class Venues
    {
        [Column(IsPrimaryKey = true)]
        public string name { get; set; }
        [Column]
        public string t { get; set; }
        [Column(CanBeNull=true)]
        public string imageUri { get; set; }
        [Column(CanBeNull=true)]
        public string description_short { get; set; }
        [Column(CanBeNull = true)]
        public string description { get; set; }
        [Column(CanBeNull = true)]
        public string webaddress { get; set; }
        [Column]
        public double latitude { get; set; }
        [Column]
        public double longitude { get; set; }
    }
}
```

CustomPushPin.cs

```
using System;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Ink;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Controls.Maps;
using System.Device.Location;
using Portsmouth_Guide.Model;
using System.ComponentModel;
using System.Collections.ObjectModel;
using Microsoft.Phone.Shell;
using System.Windows.Navigation;

namespace Portsmouth_Guide
{
    public class CustomPushPin
    {
        private string _content;
        public string Content
        {
            get { return _content; }
            set { _content = value; }
        }

        private GeoCoordinate _location;
```

```
        public GeoCoordinate Location
        {
            get { return _location; }
            set { _location = value; }
        }

        private string _typeOfPin;
        public string TypeOfPin
        {
            get { return _typeOfPin; }
            set { _typeOfPin = value; }
        }

    }
}
```

MainPage.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
            xmlns:i="clr-
namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity" xmlns:ec="clr-
namespace:Microsoft.Expression.Interactivity.Core;assembly=Microsoft.Expression.Interactions
"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.MainPage"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            SupportedOrientations="Portrait" Orientation="Portrait"
            shell:SystemTray.IsVisible="True">

            <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="images\mainbackground.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--TitlePanel contains the name of the application and page title-
->
                        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
                                    <TextBlock x:Name="PageTitle" Margin="9,-7,0,0"
Style="{StaticResource PhoneTextTitle1Style}" Text="Portsmouth" FontFamily="Cambria"/>
                        </StackPanel>

                        <!--ContentPanel - place additional content here-->
                        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <Rectangle x:Name="locationRect" Height="67" Margin="30,30,30,0"
Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295" StrokeThickness="5"
RenderTransformOrigin="0,0" RadiusY="30" RadiusX="30">
                                                <Rectangle.Fill>
                                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
```

```
                                                                <GradientStop
Color="Black" Offset="0"/>
                                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                                            </LinearGradientBrush>
                                                        </Rectangle.Fill>
                                    </Rectangle>
                                    <Rectangle x:Name="uniRect" Height="67"
Margin="30,105,30,0" Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295"
StrokeThickness="5" RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30">
                                                        <Rectangle.Fill>
                                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                                <GradientStop
Color="Black" Offset="0"/>
                                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                                            </LinearGradientBrush>
                                                        </Rectangle.Fill>
                                    </Rectangle>
                                    <Rectangle x:Name="restRect" Height="67"
Margin="30,180,30,0" Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295"
StrokeThickness="5" RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30">
                                                        <Rectangle.Fill>
                                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                                <GradientStop
Color="Black" Offset="0"/>
                                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                                            </LinearGradientBrush>
                                                        </Rectangle.Fill>
                                    </Rectangle>
                                    <Rectangle x:Name="musRect" Height="67"
Margin="30,255,30,0" Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295"
StrokeThickness="5" RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30">
                                                        <Rectangle.Fill>
                                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                                <GradientStop
Color="Black" Offset="0"/>
                                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                                            </LinearGradientBrush>
                                                        </Rectangle.Fill>
                                    </Rectangle>
                                    <Rectangle x:Name="theaRect" Margin="30,330,30,0"
Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295" StrokeThickness="5"
RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30" Height="67">
                                                        <Rectangle.Fill>
                                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                                <GradientStop
Color="Black" Offset="0"/>
                                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                                            </LinearGradientBrush>
                                                        </Rectangle.Fill>
                                    </Rectangle>
                                    <Rectangle x:Name="shopRect" Height="67"
Margin="30,405,30,0" Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295"
StrokeThickness="5" RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30">
```

```
                                        <Rectangle.Fill>
                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                <GradientStop
Color="Black" Offset="0"/>
                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                            </LinearGradientBrush>
                                        </Rectangle.Fill>
            </Rectangle>
                                    <Rectangle x:Name="aboutRect" Height="67"
Margin="30,480,30,0" Stroke="#FF0000F9" VerticalAlignment="Top" Opacity="0.295"
StrokeThickness="5" RenderTransformOrigin="0,0" RadiusX="30" RadiusY="30">
                                        <Rectangle.Fill>
                                            <LinearGradientBrush
EndPoint="0.5,1" StartPoint="0.5,0">
                                                <GradientStop
Color="Black" Offset="0"/>
                                                <GradientStop
Color="#FF265080" Offset="1"/>
                                            </LinearGradientBrush>
                                        </Rectangle.Fill>
            </Rectangle>
            <TextBlock x:Name="locText" Height="35" Margin="76,45,80,0" TextWrapping="Wrap"
Text="Location" VerticalAlignment="Top" FontFamily="Cambria" FontSize="26.667"
RenderTransformOrigin="0.5,0.5" FontWeight="Bold" TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
                                    <TextBlock x:Name="uniText" Height="35"
Margin="76,120,60,0" TextWrapping="Wrap" Text="University of Portsmouth"
VerticalAlignment="Top" FontFamily="Cambria" FontSize="26.667"
RenderTransformOrigin="0.5,0.5" FontWeight="Bold" TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
                                    <TextBlock x:Name="restText" Height="35"
Margin="76,195,80,0" TextWrapping="Wrap" Text="Food &amp;amp; Drink" VerticalAlignment="Top"
FontFamily="Cambria" FontSize="26.667" RenderTransformOrigin="0.5,0.5" FontWeight="Bold"
TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
                                    <TextBlock x:Name="musText" Height="35"
Margin="76,270,80,0" TextWrapping="Wrap" Text="Museums &amp;amp; History"
VerticalAlignment="Top" FontFamily="Cambria" FontSize="26.667"
RenderTransformOrigin="0.5,0.5" FontWeight="Bold" TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
                                    <TextBlock x:Name="theaText" Height="35"
Margin="76,345,80,0" TextWrapping="Wrap" Text="Theatres &amp;amp; Cinema"
VerticalAlignment="Top" FontFamily="Cambria" FontSize="26.667"
RenderTransformOrigin="0.5,0.5" FontWeight="Bold" TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
```

```
                                    <TextBlock x:Name="shopText" Height="35"
Margin="76,420,80,0" TextWrapping="Wrap" Text="Shopping" VerticalAlignment="Top"
FontFamily="Cambria" FontSize="26.667" RenderTransformOrigin="0.5,0.5" FontWeight="Bold"
TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
                                    <TextBlock x:Name="aboutText" Height="35"
Margin="76,495,80,0" TextWrapping="Wrap" Text="About Portsmouth" VerticalAlignment="Top"
FontFamily="Cambria" FontSize="26.667" RenderTransformOrigin="0.5,0.5" FontWeight="Bold"
TextAlignment="Center">
                                        <TextBlock.RenderTransform>
                                            <CompositeTransform/>
                                        </TextBlock.RenderTransform>
                                    </TextBlock>
            <Rectangle x:Name="locClickRect" Height="75" Margin="30,26,30,0" Stroke="Black"
StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction TargetPage="/Screens/LocationScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="uniClickRect" Height="75" Margin="30,101,30,0" Stroke="Black"
StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction
TargetPage="/Screens/UniversityScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="restClickRect" Height="75" Margin="30,176,30,0"
Stroke="Black" StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction
TargetPage="/Screens/RestaurantsScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="musClickRect" Height="75" Margin="30,251,30,0" Stroke="Black"
StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction TargetPage="/Screens/MuseumsScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="theaClickRect" Height="75" Margin="30,326,30,0"
Stroke="Black" StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction TargetPage="/Screens/TheatresScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="shopClickRect" Height="75" Margin="30,401,30,0"
Stroke="Black" StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
```

```
                        <ec:NavigateToPageAction TargetPage="/Screens/ShoppingScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Rectangle x:Name="aboutClickRect" Height="75" Margin="30,476,30,0"
Stroke="Black" StrokeThickness="5" VerticalAlignment="Top" Opacity="0" Fill="Black">
                <i:Interaction.Triggers>
                    <i:EventTrigger EventName="MouseLeftButtonDown">
                        <ec:NavigateToPageAction TargetPage="/Screens/AboutScreen.xaml"/>
                    </i:EventTrigger>
                </i:Interaction.Triggers>
            </Rectangle>
            <Button x:Name="addBtn" Content="A" HorizontalAlignment="Left" Height="86"
VerticalAlignment="Bottom" Width="112" Background="Transparent" BorderBrush="#FF10217E"
Foreground="#FF10217E" FontFamily="Segoe Keycaps" FontSize="48" FontWeight="Bold"
BorderThickness="0" Margin="-28,0,0,0" Click="addBtn_Click"/>
        </Grid>
            </Grid>
</phone:PhoneApplicationPage>
```

MainPage.xaml.cs

```
namespace Portsmouth_Guide
{
    public partial class MainPage : PhoneApplicationPage
    {
        public MainPage()
        {
            InitializeComponent();

        }

        private void addBtn_Click(object sender, RoutedEventArgs e)
        {
            NavigationService.Navigate(new Uri("/Screens/AddVenueScreen.xaml",
UriKind.Relative));
        }
    }
}
```

AddVenueScreen.xaml

```
<phone:PhoneApplicationPage
    x:Class="Portsmouth_Guide.Screens.AddVenueScreen"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    mc:Ignorable="d" d:DesignHeight="768" d:DesignWidth="480"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot is the root grid where all page content is placed-->
    <Grid x:Name="LayoutRoot">
```

```
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!--TitlePanel contains the name of the application and page title-->
        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
            <TextBlock x:Name="PageTitle" Text="Add New Venue" Margin="9,-7,0,0"
Style="{StaticResource PhoneTextTitle1Style}" FontFamily="Cambria" FontSize="48"/>
        </StackPanel>

        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <TextBlock Height="52" Margin="9,6,336,0" Name="latLabel" Text="Latitude:"
VerticalAlignment="Top" HorizontalAlignment="Right" />
            <TextBlock Height="52" Margin="6,64,336,0" Name="longLabel" Text="Longitude:"
VerticalAlignment="Top" HorizontalAlignment="Right" />
            <TextBox Height="90" HorizontalAlignment="Left" Margin="9,122,0,0"
Name="nameOfLoc" Text="Add name here..." VerticalAlignment="Top" Width="402" />
            <TextBlock Height="50" HorizontalAlignment="Left" Margin="126,8,0,0"
Name="latitude" Text="" VerticalAlignment="Top" Width="264" />
            <TextBlock Height="50" HorizontalAlignment="Left" Margin="126,64,0,0"
Name="longitude" Text="" VerticalAlignment="Top" Width="264" />
            <Button Content="Get Location" Height="92" HorizontalAlignment="Left"
Margin="6,424,0,0" Name="showLoc" VerticalAlignment="Top" Width="225" Click="showLoc_Click"
/>
            <Button Content="Save Location" Height="92" HorizontalAlignment="Right"
Margin="0,424,0,0" Name="saveLoc" VerticalAlignment="Top" Width="225" Click="saveLoc_Click"
/>
            <Button Content="List Locations" Height="92" HorizontalAlignment="Left"
Margin="9,522,0,0" Name="listLoc" VerticalAlignment="Top" Width="225" Click="listLoc_Click"
/>
            <ListBox
                Height="200"
                FontSize="25"
                Margin="26,218,0,0"
                Name="listOfTypes"
                VerticalAlignment="Top"
                HorizontalAlignment="Left"
                Width="275"
                SelectionMode="Extended">
                <ListBoxItem>Restraurant</ListBoxItem>
                <ListBoxItem>Pub</ListBoxItem>
                <ListBoxItem>Coffee</ListBoxItem>
                <ListBoxItem>Theater</ListBoxItem>
                <ListBoxItem>History</ListBoxItem>
                <ListBoxItem>Fashion</ListBoxItem>
                <ListBoxItem>Sport</ListBoxItem>
                <ListBoxItem>University</ListBoxItem>
            </ListBox>
        </Grid>
    </Grid>

    <!--Sample code showing usage of ApplicationBar-->
    <!--<phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="True">
            <shell:ApplicationBarIconButton IconUri="/Images/appbar_button1.png"
Text="Button 1"/>
```

```
            <shell:ApplicationBarIconButton IconUri="/Images/appbar_button2.png"
Text="Button 2"/>
            <shell:ApplicationBar.MenuItems>
                <shell:ApplicationBarMenuItem Text="MenuItem 1"/>
                <shell:ApplicationBarMenuItem Text="MenuItem 2"/>
            </shell:ApplicationBar.MenuItems>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>-->

</phone:PhoneApplicationPage>
```

AddVenueScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Controls.Maps;
using System.Device.Location;
using System.Collections.ObjectModel;
using System.IO;
using System.IO.IsolatedStorage;


namespace Portsmouth_Guide.Screens
{
    public partial class AddVenueScreen : PhoneApplicationPage
    {

        GeoCoordinateWatcher watcher2;
        String tempName;
        String tempType;
        String tempString;
        double tempLatitude;
        double tempLongitude;

        IsolatedStorageFile myStorage;
        StreamWriter writeToFile;

        private List<String> _listOfLocations = new List<String>();
        public List<String> ListOfLocations
        {
            get { return _listOfLocations; }
            set { _listOfLocations = value; }
        }

        public AddVenueScreen()
        {
            InitializeComponent();

            try
            {
                myStorage = IsolatedStorageFile.GetUserStoreForApplication();
                if (!myStorage.FileExists("locations.txt"))
```

```
                {
                    writeToFile = new StreamWriter(new
IsolatedStorageFileStream("locations.txt", FileMode.Create, myStorage));
                    writeToFile.Close();
                }
                IsolatedStorageFileStream myStream = new
IsolatedStorageFileStream("locations.txt", FileMode.Open, FileAccess.Read, myStorage);

                using (StreamReader reader = new StreamReader(myStream))
                {
                    int i = 0;
                    while ((tempString = reader.ReadLine()) != null)
                    {
                        ListOfLocations.Insert(i, tempString);
                        i++;
                    }
                    reader.Close();
                }
            }
            catch (Exception e)
            {
                MessageBox.Show("Isolated Storage not working!");
            }

        }

        private void showLoc_Click(object sender, RoutedEventArgs e)
        {
            if (watcher2 == null)
            {
                watcher2 = new GeoCoordinateWatcher(GeoPositionAccuracy.High);
                watcher2.MovementThreshold = 10;
                watcher2.StatusChanged += new
EventHandler<GeoPositionStatusChangedEventArgs>(watcher2_StatusChanged);
                watcher2.PositionChanged += new
EventHandler<GeoPositionChangedEventArgs<GeoCoordinate>>(watcher2_PositionChanged);
            }
            watcher2.Start();
        }

        void watcher2_StatusChanged(object sender, GeoPositionStatusChangedEventArgs e)
        {

        }

        void watcher2_PositionChanged(object sender,
GeoPositionChangedEventArgs<GeoCoordinate> e)
        {
            latitude.Text = e.Position.Location.Latitude.ToString("0.00000");
            tempLatitude = e.Position.Location.Latitude;
            longitude.Text = e.Position.Location.Longitude.ToString("0.00000");
            tempLongitude = e.Position.Location.Longitude;
        }

        private void saveLoc_Click(object sender, RoutedEventArgs e)
        {
            if (listOfTypes.SelectedItem.ToString() != null)
            {
                tempName = nameOfLoc.Text;
                ListBoxItem selection = (ListBoxItem)listOfTypes.SelectedItem;
                tempType = selection.Content.ToString();
```

```
                tempString = tempName + " " + tempType + " " + tempLatitude.ToString() + " "
+ tempLongitude.ToString();
                var newVenue = new Venues { name = tempName, t = tempType, imageUri = null,
description_short = null, description = null, webaddress = null, latitude = tempLatitude,
longitude = tempLongitude };
                if (ListOfLocations.Count > 0)
                {
                    ListOfLocations.Insert(ListOfLocations.Count, tempString);
                }
                else
                {
                    ListOfLocations.Add(tempString);
                }
                using (var file = new IsolatedStorageFileStream("locations.txt",
FileMode.OpenOrCreate, FileAccess.Write, myStorage))
                {
                    using (writeToFile = new StreamWriter(file))
                    {
                        foreach (string data in ListOfLocations)
                        {
                            writeToFile.WriteLine(data);
                        }
                    }
                }
            }
            else { MessageBox.Show("Please select the type of venue."); }

            nameOfLoc.Text = "Add Name Here...";
            latitude.Text = "";
            longitude.Text = "";

        }

        private void listLoc_Click(object sender, RoutedEventArgs e)
        {
            NavigationService.Navigate(new Uri("/Screens/ShowNewListScreen.xaml",
UriKind.Relative));
        }
    }
}
```

InfoScreen.xaml

```
<phone:PhoneApplicationPage
    x:Class="Portsmouth_Guide.Screens.InfoScreen"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    mc:Ignorable="d" d:DesignHeight="696" d:DesignWidth="480"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot is the root grid where all page content is placed-->
    <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
```

```
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!--TitlePanel contains the name of the application and page title-->
        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,12,0,12">
            <TextBlock x:Name="pageTitle" Text="-" VerticalAlignment="Top" Height="68"
TextWrapping="Wrap" FontSize="40"/>
        </StackPanel>

        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <TextBlock x:Name="typeTitle" Text="-" FontSize="20" Height="35"
VerticalAlignment="TOP" Margin="0,-23,0,0" />
        </Grid>
        <TextBlock x:Name="descrLabel" Height="64" Margin="12,44,204,0" Grid.Row="1"
TextWrapping="Wrap" Text="Description" VerticalAlignment="Top" FontSize="32"
FontWeight="Bold"/>
        <TextBlock x:Name="description" Height="188" Margin="12,112,12,0" Grid.Row="1"
TextWrapping="Wrap" Text="-" VerticalAlignment="Top" FontSize="21.333"/>
        <TextBlock x:Name="webaddressLabel" Height="64" Margin="12,300,184,208" Grid.Row="1"
TextWrapping="Wrap" Text="Website" RenderTransformOrigin="0.561,2.625" FontSize="32"
FontWeight="Bold"/>
        <Image x:Name="buildingImg"
               Height="auto"
               Margin="12,0,12,12"
               Grid.Row="1"
               VerticalAlignment="Bottom"
               HorizontalAlignment="Left"
               Source="{Binding imgUri}"
               Visibility="Collapsed"
               MaxHeight="320"/>
        <HyperlinkButton
            x:Name="webaddressLink"
            Height="64"
            Margin="12,0,12,185"
            Grid.Row="1"
            VerticalAlignment="Bottom"
            HorizontalAlignment="Left"
            NavigateUri="{Binding webaddr}"
            Content="-"
            TargetName="_blank"></HyperlinkButton>
    </Grid>

    <!--Sample code showing usage of ApplicationBar-->
    <phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="False">
            <shell:ApplicationBarIconButton x:Name="LocationBtn"
IconUri="/images/appbarPosition.png" Text="Location" Click="LocationBtn_Click"/>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>

</phone:PhoneApplicationPage>
```

InfoScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
```

```
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using Portsmouth_Guide.Model;
using Portsmouth_Guide.ViewModel;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using System.Collections.ObjectModel;
using System.Windows.Media.Imaging;

namespace Portsmouth_Guide.Screens
{
    public partial class InfoScreen : PhoneApplicationPage
    {
        string dataStream;
        string name;
        string type;
        string descr_short;
        string descr;
        string webaddr;
        double latitude;
        double longitude;

        Uri imgUri;

        public InfoScreen()
        {
            InitializeComponent();
        }

        protected override void OnNavigatedTo(NavigationEventArgs e)
        {
            base.OnNavigatedTo(e);

            dataStream = this.NavigationContext.QueryString["name"];

            AppDataContext dc = new AppDataContext("Data Source=isostore:/appData.sdf");

            Table<Venues> Venue = dc.GetTable<Venues>();

            var data = from d in dc.Venues
                       where d.name == dataStream
                       select d;

            foreach (var da in data)
            {
                name = da.name;
                type = da.t;
                descr_short = da.description_short;
                descr = da.description;
                webaddr = da.webaddress;
                latitude = da.latitude;
                longitude = da.longitude;
                imgUri = new Uri(da.imageUri, UriKind.RelativeOrAbsolute);
            }

            if (type == "University")
```

```
            {
                descrLabel.Text = "Address";
                ImageSource imgSource = new BitmapImage(imgUri);
                this.buildingImg.Source = imgSource;
                webaddressLabel.Visibility = Visibility.Collapsed;
                webaddressLink.Visibility = Visibility.Collapsed;
                buildingImg.Visibility = Visibility.Visible;
            }

            this.pageTitle.Text = name;
            this.typeTitle.Text = type;
            description.Text = descr;
            webaddressLink.Content = webaddr;
        }

        private void LocationBtn_Click(object sender, EventArgs e)
        {
            NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", latitude, longitude),
                UriKind.RelativeOrAbsolute));
        }
    }
}
```

LocationScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:my="clr-
namespace:Microsoft.Phone.Controls.Maps;assembly=Microsoft.Phone.Controls.Maps"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="696"
            x:Class="Portsmouth_Guide.LocationPage"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True">


            <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--TitlePanel contains the name of the application and page title-
->
                        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
                                    <TextBlock x:Name="PageTitle" Text="Location Services"
Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}" FontSize="48"
FontFamily="Cambria"/>
                        </StackPanel>
```

```
                        <!--ContentPanel - place additional content here-->
                        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <my:Map Height="532" Mode="Road" HorizontalAlignment="Left" Name="map1"
VerticalAlignment="Top"
                Width="450" CopyrightVisibility="Collapsed" LogoVisibility="Collapsed"

CredentialsProvider="AuCbUj4yEfpfjzjamE62Ah3UHwuzlcoVSj8V4OvceAWwsu7xjgSo9vRkAzHmWPWO"
Margin="6,0,0,0">
                <my:MapItemsControl ItemsSource="{Binding Pushpins}">
                    <my:MapItemsControl.ItemTemplate>
                        <DataTemplate>
                            <my:Pushpin Location="{Binding Location}">
                                <Image x:Name="pinImg" Source="{Binding pushpinImg}"
Height="40" Width="20" />
                            </my:Pushpin>
                        </DataTemplate>
                    </my:MapItemsControl.ItemTemplate>
                </my:MapItemsControl>
            </my:Map>
            <TextBlock Height="50" HorizontalAlignment="Center" Margin="10,0,0,13"
Name="statusOfTracking" FontSize="24" Text="To start, press Find me! (the crosshair)"
VerticalAlignment="Bottom" Width="auto"/>
        </Grid>
    </Grid>
    <!--Sample code showing usage of ApplicationBar-->
    <phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="True">
            <shell:ApplicationBarIconButton IconUri="/images/appbarPosition.png" Text="Find
Me!" x:Name="trackMe" Click="trackMe_Click" />
            <shell:ApplicationBarIconButton IconUri="/images/appbarRefresh.png" Text="What's
close" x:Name="updateMap" Click="updateMap_Click"/>
            <shell:ApplicationBarIconButton IconUri="/images/appbarSettings.png"
Text="Aerial" x:Name="viewMode" Click="viewMode_Click"/>
            <shell:ApplicationBar.MenuItems>

            </shell:ApplicationBar.MenuItems>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>
</phone:PhoneApplicationPage>
```

LocationScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Controls.Maps;
using System.Device.Location;
using System.Threading;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using Portsmouth_Guide.Model;
using System.ComponentModel;
using System.Collections.ObjectModel;
```

```
using Microsoft.Phone.Shell;
using System.Windows.Navigation;
using System.Windows.Media.Imaging;



namespace Portsmouth_Guide
{
    public partial class LocationPage : PhoneApplicationPage
    {
        GeoCoordinateWatcher watcher;
        bool trackingOn;
        Pushpin myLocation;
        CustomPushPin venueLoc;
        ObservableCollection<CustomPushPin> Pushpins = new
ObservableCollection<CustomPushPin>();
        List<GeoCoordinate> ListOfLoc = new List<GeoCoordinate>();

        MapLayer pushpinLayer;
        Image pushpinImg;

        double latitudeStream;
        double longitudeStream;

        public LocationPage()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;

            watcher = new GeoCoordinateWatcher(GeoPositionAccuracy.Default); // new watcher
for position with high accuracy
            watcher.MovementThreshold = 10.0f; // Update the location every 10 meters
            trackingOn = false; // let's not track the movement before the user wants to, to
save data and battery life.
            myLocation = new Pushpin(); // PushPin to show the current location
            venueLoc = new CustomPushPin();

            watcher.StatusChanged += new
EventHandler<GeoPositionStatusChangedEventArgs>(watcher_StatusChanged);
            watcher.PositionChanged += new
EventHandler<GeoPositionChangedEventArgs<GeoCoordinate>>(watcher_PositionChanged);

        }

        // Let's check whether the status of the watcher changes and then inform the user
        // by inputting text in the statusOfTracking textBox
        void watcher_StatusChanged(object sender, GeoPositionStatusChangedEventArgs e)
        {
            switch (e.Status)
            {
                case GeoPositionStatus.Ready: // Ready, working and receiving data.
                    statusOfTracking.Text = "Location acquired.";
                    break;
                case GeoPositionStatus.Initializing: // Still initializing and getting data
                    statusOfTracking.Text = "Retrieving data...";
                    break;
                case GeoPositionStatus.NoData: // Working but can't get location data
                    statusOfTracking.Text = "No location data available...";
                    break;
                case GeoPositionStatus.Disabled: //Location service disabled or unsupported,
let's do a quick check.
```

```
                    if (watcher.Permission == GeoPositionPermission.Denied)
                    {
                        statusOfTracking.Text = "Your location service is disabled.";
                    }
                    else
                    {
                        statusOfTracking.Text = "Location service doesn't work on this
device.";
                    }
                    break;
            }
        }

        //If the position is changed, check if tracking is On
        //Add an image to the pushpin and add the location to a list
        //Create a rectangle of all the pushpin locations and set the map around that
        //Check that the zoom isn't too close
        //Set the height and width of the Pushpin image
        //Create a new instance of pushpinLayer and add that layer on to the map
        void watcher_PositionChanged(object sender,
GeoPositionChangedEventArgs<GeoCoordinate> e)
        {
            if (trackingOn)
            {
                myLocation.Location = e.Position.Location;
                pushpinImg = new Image();
                pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_5.png",
UriKind.Relative));
                ListOfLoc.Add(myLocation.Location);
                LocationRect locrect = LocationRect.CreateLocationRect(ListOfLoc);
                map1.SetView(locrect);
                if (map1.ZoomLevel > 13.0f) { map1.ZoomLevel = 13.0f; }
                pushpinImg.Height = 40;
                pushpinImg.Width = 20;
                pushpinLayer = new MapLayer();
                pushpinLayer.AddChild(pushpinImg, myLocation.Location);
                if (map1.Children.Contains(pushpinLayer) == false) {
map1.Children.Add(pushpinLayer); }

            }
        }

        //In this startLocService method we initialize the watcher and give it 30seconds,
before it times out if unsuccesful
        //By threading this start, we don't hold up the whole application for the duration
of the start
        void startLocService()
        {
            watcher.TryStart(true, TimeSpan.FromMilliseconds(30000)); //30 seconds for the
initialization
        }

        //When the user presses trackMe button we should either start or stop tracking
        private void trackMe_Click(object sender, EventArgs e)
        {
            if (trackingOn) // If tracking is already on and button pressed
            {
                trackingOn = false; // Tracking OFF
                map1.ZoomLevel = 1.0f; // Zoom out
                statusOfTracking.Text = "Location Service Stopped..."; // Change the text on
status textbox
                watcher.Stop(); // Stop watcher
```

```
            }
            else
            { // If tracking is off and button pressed
                trackingOn = true; // Tracking ON
                map1.ZoomLevel = 16.0f; // Zoom in
                statusOfTracking.Text = "Location Service Starting..."; // Change the text
on status textbox
                new Thread(startLocService).Start(); // Call startLocService method
            }
        }

        //When the users checks the surrounding area for venues, check that the tracking is
on
        //Call for the loadVenues() and layoutPushPins() methods
        //Create a location rectangle and set the map to zoom around that rectangle
        private void updateMap_Click(object sender, EventArgs e)
        {
            if (trackingOn)
            {
                loadVenues();
                layoutPushPins();

                LocationRect locrect = LocationRect.CreateLocationRect(ListOfLoc);
                map1.SetView(locrect);
            }
        }

        //Load all the venues from the database one by one
        //Create a new instance of our CustomPushPin and set it's name, type and location
values
        //Add the new Pushpin to the pushpin- and pushpinlocation lists
        private void loadVenues()
        {
            foreach (var data in App.ViewModel.AllVenues)
            {
                if (data.t != "University")
                {
                    venueLoc = new CustomPushPin();
                    venueLoc.Content = data.name;
                    venueLoc.TypeOfPin = data.t;

                    GeoCoordinate tempLoc = new GeoCoordinate(data.latitude,
data.longitude);
                    venueLoc.Location = tempLoc;
                    ListOfLoc.Add(tempLoc);
                    Pushpins.Add(venueLoc);
                }
            }
        }

        //When the users wants to change the viewmode, change the mode of the map and the
text on the button
        private void viewMode_Click(object sender, EventArgs e)
        {
            ApplicationBarIconButton btn =
(ApplicationBarIconButton)ApplicationBar.Buttons[2];
            if (btn.Text == "Aerial")
            {
                map1.Mode = new AerialMode(true);
                btn.Text = "Road";
            }
```

```
            else {
                map1.Mode = new RoadMode();
                btn.Text = "Aerial";
            }
        }

        //Override method for the OnNavigatedTo, that handles the incoming data from other
screens
        protected override void OnNavigatedTo(NavigationEventArgs e)
        {
            base.OnNavigatedTo(e);
            latitudeStream = 0.0;
            longitudeStream = 0.0;

            //Check if data is sent with the navigation service, and especially if the data
is named as "lat"
            if (this.NavigationContext.QueryString.ContainsKey("lat"))
            {
                Pushpins.Clear();   //Clear the Pushpin collection
                latitudeStream =
Convert.ToDouble(this.NavigationContext.QueryString["lat"]);   //Receive and convert the
incoming string
                longitudeStream =
Convert.ToDouble(this.NavigationContext.QueryString["long"]); //to latitudeStream and
longitudeStream double values

                AppDataContext dc = new AppDataContext("Data Source=isostore:/appData.sdf");
//Connect to the database

                Table<Venues> Venue = dc.GetTable<Venues>();    //Get Table Venues from the
database

                var data = from d in dc.Venues                  //Check which venue in the
database is located in the coordinates and select it
                           where d.latitude == latitudeStream
                           && d.longitude == longitudeStream
                           select d;

                //Create a new instance of CustomPushPin and add the name and type value
from the venue queried before
                venueLoc = new CustomPushPin();
                foreach (var da in data)
                {
                    venueLoc.Content = da.name;
                    venueLoc.TypeOfPin = da.t;
                }

                GeoCoordinate tempLoc = new GeoCoordinate(latitudeStream, longitudeStream);
                venueLoc.Location = tempLoc;
                ListOfLoc.Add(tempLoc);
                Pushpins.Add(venueLoc);

                trackingOn = true; // Tracking ON
                statusOfTracking.Text = "Location Service Starting..."; // Change the text
on status textbox
                new Thread(startLocService).Start(); // Call startLocService method

                layoutPushPins();   //Call layoutPushPins()
            }
        }
```

```
        //LayoutPushPins method goes through all the Pushpins in the collection and checks
their type value to choose the correct pushpin image
        //Pushpin image is resized and the pushpin is added to the pushpinLayer, which is
then added to the map
        public void layoutPushPins()
        {
            foreach (var p in Pushpins)
            {
                switch (p.TypeOfPin)
                {
                    case "Restaurant":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_4.png",
UriKind.Relative));
                        break;
                    case "Pub":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_4.png",
UriKind.Relative));
                        break;
                    case "Coffee":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_4.png",
UriKind.Relative));
                        break;
                    case "History":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_2.png",
UriKind.Relative));
                        break;
                    case "Fashion":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_1.png",
UriKind.Relative));
                        break;
                    case "Sport":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_1.png",
UriKind.Relative));
                        break;
                    case "Theatre":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_3.png",
UriKind.Relative));
                        break;
                    case "University":
                        pushpinImg = new Image();
                        pushpinImg.Source = new BitmapImage(new Uri("/images/Pin_5.png",
UriKind.Relative));
                        break;
                    default:
                        MessageBox.Show("Error in getting the pushpin image source");
                        break;
                }
                pushpinImg.Height = 40;
                pushpinImg.Width = 20;
                pushpinLayer = new MapLayer();
                pushpinLayer.AddChild(pushpinImg, p.Location);
                if (map1.Children.Contains(pushpinLayer) == false) {
map1.Children.Add(pushpinLayer); }
            }
        }
```

```
    }
}
```

MuseumsScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:controls="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls"
    xmlns:i="clr-
namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity"
    xmlns:ec="clr-
namespace:Microsoft.Expression.Interactivity.Core;assembly=Microsoft.Expression.Interactions
"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.Museums"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True">

            <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--TitlePanel contains the name of the application and page title-
->
                        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
                                    <TextBlock x:Name="PageTitle" Text="Museums &amp;
History" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}" FontSize="48"
FontFamily="Cambria"/>
                        </StackPanel>

                        <!--ContentPanel - place additional content here-->
                        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <ListBox x:Name="HistoryListBox"
                     ItemsSource="{Binding HistoryPlaces}"
                     Margin="12,0,12,0">
                <ListBox.ItemTemplate>
                    <DataTemplate>
                        <Grid HorizontalAlignment="Stretch" Width="420">
                            <Grid.RowDefinitions>
                                <RowDefinition Height="50" />
                                <RowDefinition Height="10" />
                                <RowDefinition Height="60" />
                            </Grid.RowDefinitions>
                            <Grid.ColumnDefinitions>
```

```
                        <ColumnDefinition Width="344" />
                        <ColumnDefinition Width="60" />
                        <ColumnDefinition Width="auto" />
                    </Grid.ColumnDefinitions>

                    <TextBlock
                        x:Name="hisName"
                        Text="{Binding name}"
                        VerticalAlignment="Top"
                        FontSize="25"
                        FontStyle="Italic"
                        Grid.Column="0"
                        Grid.Row="0"
                        Margin="10,10,0,0"
                        Tap="hisDescr_Tap"/>

                    <TextBlock
                        x:Name="hisDescr"
                        Text="{Binding description_short}"
                        TextWrapping="Wrap"
                        Grid.Row="2"
                        Grid.ColumnSpan="2"
                        Margin="4,4,0,0"
                        HorizontalAlignment="Left"
                        VerticalAlignment="top"
                        Tap="hisDescr_Tap"/>

                    <Rectangle
                        Grid.Column="1"
                        Grid.Row="0"
                        Height="50"
                        x:Name="hisRect1"
                        Stretch="Fill"
                        Width="50"
                        Fill="#ffcc00"
                        Tap="hisImg1_Tap"
                        Grid.RowSpan="2" />

                    <Image
                        Grid.Column="1"
                        Grid.Row="0"
                        Grid.RowSpan="2"
                        Height="50"
                        Width="50"
                        x:Name="hisImg1"
                        HorizontalAlignment="Center"
                        Stretch="Fill"
                        Source="/images/appbarPosition.png"
                        Tap="hisImg1_Tap"
                        />

                </Grid>
            </DataTemplate>
        </ListBox.ItemTemplate>
    </ListBox>
                </Grid>
    </Grid>
</phone:PhoneApplicationPage>
```

MuseumsScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using Portsmouth_Guide.Model;


namespace Portsmouth_Guide
{
    public partial class Museums : PhoneApplicationPage
    {
        public Museums()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;

        }
        private void hisImg1_Tap(object sender, GestureEventArgs e)
        {
            if (HistoryListBox.SelectedIndex != -1)
            {
                var dataOfPlace = HistoryListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            HistoryListBox.SelectedIndex = -1;
        }

        private void hisDescr_Tap(object sender, GestureEventArgs e)
        {
            if (HistoryListBox.SelectedIndex != -1)
            {
                var dataOfPlace = HistoryListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            HistoryListBox.SelectedIndex = -1;
        }
    }
}
```

RestaurantsScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
```

```
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
            xmlns:controls="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls" xmlns:i="clr-
namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity" xmlns:ec="clr-
namespace:Microsoft.Expression.Interactivity.Core;assembly=Microsoft.Expression.Interactions
"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.RestaurantsScreen"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True">

    <phone:PhoneApplicationPage.Resources>
        <DataTemplate x:Key="DiningListBoxItemTemplate">
            <Grid HorizontalAlignment="Stretch" Width="420">
                <Grid.RowDefinitions>
                    <RowDefinition Height="50" />
                    <RowDefinition Height="10" />
                    <RowDefinition Height="60" />
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="344" />
                    <ColumnDefinition Width="60" />
                    <ColumnDefinition Width="auto" />
                </Grid.ColumnDefinitions>

                <TextBlock
                    Grid.Column="0"
                    Grid.Row="0"
                    Margin="10,10,0,0"
                    x:Name="resName"
                    Text="{Binding name}"
                    VerticalAlignment="Top"
                    FontSize="25"
                    FontStyle="Italic"
                    Tap="resDescr_Tap"/>

                <TextBlock
                    Grid.ColumnSpan="2"
                    Grid.Row="2"
                    HorizontalAlignment="Left"
                    Margin="4,4,0,0"
                    x:Name="resDescr"
                    Text="{Binding description_short}"
                    VerticalAlignment="top"
                    TextWrapping="Wrap"
                    Tap="resDescr_Tap"/>

                <Rectangle
                    Grid.Column="1"
                    Grid.Row="0"
                    Height="50"
                    x:Name="resRect1"
                    Stretch="Fill"
                    Width="60"
                    Fill="#00ff0c"
                    Tap="resImg1_Tap"
```

```
                    Grid.RowSpan="2" />

                <Image
                    Grid.Column="1"
                    Grid.Row="0"
                    Grid.RowSpan="2"
                    Height="50"
                    Width="50"
                    x:Name="resImg1"
                    HorizontalAlignment="Left"
                    Stretch="Fill"
                    Source="/images/appbarPosition.png"
                    Tap="resImg1_Tap"
                    />

            </Grid>
        </DataTemplate>
    </phone:PhoneApplicationPage.Resources>



                    <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--ContentPanel - place additional content here-->
                        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
                                    <controls:Pivot Margin="8" Title="Restaurants and
Pubs">
                <controls:PivotItem Header="restaurants">
                    <ListBox
                        x:Name="RestaurantsListBox"
                        ItemsSource="{Binding RestaurantPlaces}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource DiningListBoxItemTemplate}" />
                </controls:PivotItem>
                <controls:PivotItem Header="pubs">
                    <ListBox
                        x:Name="PubsListBox"
                        ItemsSource="{Binding PubPlaces}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource DiningListBoxItemTemplate}" />
                </controls:PivotItem>
                <controls:PivotItem Header="coffee">
                    <ListBox
                        x:Name="CoffeeListBox"
                        ItemsSource="{Binding CoffeePlaces}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource DiningListBoxItemTemplate}" />
                </controls:PivotItem>
                                                <controls:PivotItem Header="all">
                    <ListBox
                        x:Name="allRestaurantsAndPubsListBox"
                        ItemsSource="{Binding AllRestaurantPlaces}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource DiningListBoxItemTemplate}"/>
```

```
                </controls:PivotItem>
                                    </controls:Pivot>
        </Grid>
    </Grid>
</phone:PhoneApplicationPage>
```

RestaurantsScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using Portsmouth_Guide.Model;

namespace Portsmouth_Guide
{
    public partial class RestaurantsScreen : PhoneApplicationPage
    {
        public RestaurantsScreen()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;
        }

        private void resImg1_Tap(object sender, GestureEventArgs e)
        {
            if (RestaurantsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = RestaurantsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            RestaurantsListBox.SelectedIndex = -1;
            if (PubsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = PubsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            PubsListBox.SelectedIndex = -1;
            if (CoffeeListBox.SelectedIndex != -1)
            {
                var dataOfPlace = CoffeeListBox.SelectedItem as Venues;
```

```
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            CoffeeListBox.SelectedIndex = -1;
            if (allRestaurantsAndPubsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = allRestaurantsAndPubsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            allRestaurantsAndPubsListBox.SelectedIndex = -1;

        }

        private void resDescr_Tap(object sender, GestureEventArgs e)
        {
            if (RestaurantsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = RestaurantsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            RestaurantsListBox.SelectedIndex = -1;
            if (PubsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = PubsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            PubsListBox.SelectedIndex = -1;
            if (CoffeeListBox.SelectedIndex != -1)
            {
                var dataOfPlace = CoffeeListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            CoffeeListBox.SelectedIndex = -1;
            if (allRestaurantsAndPubsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = allRestaurantsAndPubsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            allRestaurantsAndPubsListBox.SelectedIndex = -1;
        }
    }
}
```

ShoppingScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
            xmlns:controls="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls" xmlns:i="clr-
namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity" xmlns:ec="clr-
namespace:Microsoft.Expression.Interactivity.Core;assembly=Microsoft.Expression.Interactions
"
    mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.Shopping"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True">

    <phone:PhoneApplicationPage.Resources>
        <DataTemplate x:Key="ShopsListBoxItemTemplate">
            <Grid HorizontalAlignment="Stretch" Width="420">
                <Grid.RowDefinitions>
                    <RowDefinition Height="50" />
                    <RowDefinition Height="10" />
                    <RowDefinition Height="60" />
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="344" />
                    <ColumnDefinition Width="60" />
                    <ColumnDefinition Width="auto" />
                </Grid.ColumnDefinitions>

                <TextBlock
                    Grid.Column="0"
                    Grid.Row="0"
                    Margin="10,10,0,0"
                    x:Name="shopName"
                    Text="{Binding name}"
                    VerticalAlignment="Top"
                    FontSize="25"
                    FontStyle="Italic"
                    Tap="shopDescr_Tap"/>

                <TextBlock
                    Grid.ColumnSpan="2"
                    Grid.Row="2"
                    HorizontalAlignment="Left"
                    Margin="4,4,0,0"
                    x:Name="shopDescr"
                    Text="{Binding description_short}"
                    VerticalAlignment="top"
                    TextWrapping="Wrap"
                    Tap="shopDescr_Tap"/>

                <Rectangle
                    Grid.Column="1"
                    Grid.Row="0"
                    Height="50"
```

```
                    x:Name="shopRect1"
                    Stretch="Fill"
                    Width="60"
                    Fill="#ff0000"
                    Tap="shopImg1_Tap"
                    Grid.RowSpan="2" />

                <Image
                    Grid.Column="1"
                    Grid.Row="0"
                    Height="50"
                    x:Name="shopImg1"
                    HorizontalAlignment="Left"
                    Stretch="Fill"
                    Width="50"
                    Source="/images/appbarPosition.png"
                    Grid.RowSpan="2"
                    Tap="shopImg1_Tap"/>


            </Grid>
        </DataTemplate>
    </phone:PhoneApplicationPage.Resources>

    <!--LayoutRoot is the root grid where all page content is placed-->
             <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--TitlePanel contains the name of the application and page title-
->

                        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <controls:Pivot Margin="8" Title="Shopping">
                <controls:PivotItem Header="fashion">
                    <ListBox
                        x:Name="FashionListBox"
                        ItemsSource="{Binding FashionShops}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource ShopsListBoxItemTemplate}" />
                </controls:PivotItem>
                <controls:PivotItem Header="sport">
                    <ListBox
                        x:Name="SportListBox"
                        ItemsSource="{Binding SportsShops}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource ShopsListBoxItemTemplate}" />
                </controls:PivotItem>
                <controls:PivotItem Header="all">
                    <ListBox
                        x:Name="allShopsListBox"
                        ItemsSource="{Binding AllShopPlaces}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource ShopsListBoxItemTemplate}"/>
                </controls:PivotItem>
            </controls:Pivot>
```

```
        </Grid>
    </Grid>
</phone:PhoneApplicationPage>
```

ShoppingScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using Portsmouth_Guide.Model;

namespace Portsmouth_Guide
{
    public partial class Shopping : PhoneApplicationPage
    {
        public Shopping()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;
        }

        private void shopDescr_Tap(object sender, GestureEventArgs e)
        {
            if (FashionListBox.SelectedIndex != -1)
            {
                var dataOfPlace = FashionListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            FashionListBox.SelectedIndex = -1;
            if (SportListBox.SelectedIndex != -1)
            {
                var dataOfPlace = SportListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            SportListBox.SelectedIndex = -1;
            if (allShopsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = allShopsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
```

```
            }
            allShopsListBox.SelectedIndex = -1;
        }

        private void shopImg1_Tap(object sender, GestureEventArgs e)
        {
            if (FashionListBox.SelectedIndex != -1)
            {
                var dataOfPlace = FashionListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            FashionListBox.SelectedIndex = -1;
            if (SportListBox.SelectedIndex != -1)
            {
                var dataOfPlace = SportListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            SportListBox.SelectedIndex = -1;
            if (allShopsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = allShopsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            allShopsListBox.SelectedIndex = -1;
        }

    }
}
```

ShowNewListScreen.xaml

```
<phone:PhoneApplicationPage
    x:Class="Portsmouth_Guide.Screens.ShowNewListScreen"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    mc:Ignorable="d" d:DesignHeight="768" d:DesignWidth="480"
    shell:SystemTray.IsVisible="True">

    <!--LayoutRoot is the root grid where all page content is placed-->
    <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
```

```
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>

        <!--TitlePanel contains the name of the application and page title-->
        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
            <TextBlock x:Name="PageTitle" Text="List Of New Venues" Margin="9,-7,0,0"
Style="{StaticResource PhoneTextTitle1Style}" FontFamily="Cambria" FontSize="48"/>
        </StackPanel>

        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
             <ListBox x:Name="locationsList" Margin="8,8,8,8" >
                <ListBox.ItemTemplate>
                    <DataTemplate>
                        <TextBlock Text="{Binding}" TextWrapping="Wrap"/>
                    </DataTemplate>
                </ListBox.ItemTemplate>
            </ListBox>
                        </Grid>
    </Grid>

    <!--Sample code showing usage of ApplicationBar-->
    <!--<phone:PhoneApplicationPage.ApplicationBar>
        <shell:ApplicationBar IsVisible="True" IsMenuEnabled="True">
            <shell:ApplicationBarIconButton IconUri="/Images/appbar_button1.png"
Text="Button 1"/>
            <shell:ApplicationBarIconButton IconUri="/Images/appbar_button2.png"
Text="Button 2"/>
            <shell:ApplicationBar.MenuItems>
                <shell:ApplicationBarMenuItem Text="MenuItem 1"/>
                <shell:ApplicationBarMenuItem Text="MenuItem 2"/>
            </shell:ApplicationBar.MenuItems>
        </shell:ApplicationBar>
    </phone:PhoneApplicationPage.ApplicationBar>-->

</phone:PhoneApplicationPage>
```

ShowNewListScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using System.IO.IsolatedStorage;
using System.IO;

namespace Portsmouth_Guide.Screens
{
    public partial class ShowNewListScreen : PhoneApplicationPage
    {
```

```
        string tempString;

        public ShowNewListScreen()
        {
            InitializeComponent();

            IsolatedStorageFile myStorage =
IsolatedStorageFile.GetUserStoreForApplication();
            IsolatedStorageFileStream myStream = new
IsolatedStorageFileStream("locations.txt", FileMode.Open, FileAccess.Read, myStorage);
            List<string> myLocations = new List<string>();

            using (StreamReader reader = new StreamReader(myStream))
            {
                while ((tempString = reader.ReadLine()) != null)
                {
                    myLocations.Add(tempString);
                }
                reader.Close();
            }

            locationsList.ItemsSource = myLocations;

        }
    }
}
```

TheatresScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.Theatres"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True">

            <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--TitlePanel contains the name of the application and page title-
->
                        <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
                                    <TextBlock x:Name="PageTitle" Text="Theatres &amp;
Cinema" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}" FontSize="48"
FontFamily="Cambria"/>
                        </StackPanel>
```

```
                <!--ContentPanel - place additional content here-->
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
    <ListBox x:Name="TheatreListBox"
             ItemsSource="{Binding TheatrePlaces}"
             Margin="12,0,12,0">
        <ListBox.ItemTemplate>
            <DataTemplate>
                <Grid HorizontalAlignment="Stretch" Width="420">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="50" />
                        <RowDefinition Height="10" />
                        <RowDefinition Height="60" />
                    </Grid.RowDefinitions>
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="344" />
                        <ColumnDefinition Width="60" />
                        <ColumnDefinition Width="auto" />
                    </Grid.ColumnDefinitions>

                    <TextBlock
                        x:Name="theaName"
                        Text="{Binding name}"
                        VerticalAlignment="Top"
                        FontSize="25"
                        FontStyle="Italic"
                        Grid.Column="0"
                        Grid.Row="0"
                        Margin="10,10,0,0"
                        Tap="theaDescr_Tap"/>

                    <TextBlock
                        x:Name="theaDescr"
                        Text="{Binding description_short}"
                        TextWrapping="Wrap"
                        Grid.Row="2"
                        Grid.ColumnSpan="2"
                        Margin="4,4,0,0"
                        HorizontalAlignment="Left"
                        VerticalAlignment="top"
                        Tap="theaDescr_Tap"/>

                    <Rectangle
                        Grid.Column="1"
                        Grid.Row="0"
                        Height="50"
                        x:Name="theaRect1"
                        Stretch="Fill"
                        Width="50"
                        Fill="#1200ff"
                        Tap="theaImg1_Tap"
                        Grid.RowSpan="2" />

                    <Image
                        Grid.Column="1"
                        Grid.Row="0"
                        Grid.RowSpan="2"
                        Height="50"
                        Width="50"
                        x:Name="theaImg1"
                        HorizontalAlignment="Center"
                        Stretch="Fill"
```

```
                                Source="/images/appbarPosition.png"
                                Tap="theaImg1_Tap"
                                />

                        </Grid>
                    </DataTemplate>
                </ListBox.ItemTemplate>
            </ListBox>
        </Grid>
    </Grid>
</phone:PhoneApplicationPage>
```

TheatresScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using System.Data.Linq;
using System.Data.Linq.Mapping;
using Portsmouth_Guide.Model;

namespace Portsmouth_Guide
{
    public partial class Theatres : PhoneApplicationPage
    {
        public Theatres()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;
        }

        private void theaImg1_Tap(object sender, GestureEventArgs e)
        {
            if (TheatreListBox.SelectedIndex != -1)
            {
                var dataOfPlace = TheatreListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            TheatreListBox.SelectedIndex = -1;
        }

        private void theaDescr_Tap(object sender, GestureEventArgs e)
        {
            if (TheatreListBox.SelectedIndex != -1)
            {
                var dataOfPlace = TheatreListBox.SelectedItem as Venues;
```

```
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            TheatreListBox.SelectedIndex = -1;
        }


    }
}
```

UniversityScreen.xaml

```
<phone:PhoneApplicationPage
            xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
            xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
            xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
            xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
            xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
            xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
            mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
            x:Class="Portsmouth_Guide.University"
            FontFamily="{StaticResource PhoneFontFamilyNormal}"
            FontSize="{StaticResource PhoneFontSizeNormal}"
            Foreground="{StaticResource PhoneForegroundBrush}"
            Orientation="Portrait"
            shell:SystemTray.IsVisible="True" xmlns:controls="clr-
namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone.Controls">

    <phone:PhoneApplicationPage.Resources>
        <DataTemplate x:Key="BuildingsListBoxItemTemplate">
            <Grid HorizontalAlignment="Stretch" Width="380">
                <Grid.RowDefinitions>
                    <RowDefinition Height="50" />
                    <RowDefinition Height="10" />
                    <RowDefinition Height="60" />
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="320" />
                    <ColumnDefinition Width="60" />
                </Grid.ColumnDefinitions>

                <Image
                    Grid.Column="1"
                    Grid.Row="0"
                    Height="50"
                    x:Name="resImg1"
                    Stretch="Fill"
                    Width="50"
                    Source="/images/appbarPosition.png"
                    Grid.RowSpan="2"
                    Tap="uniImg1_Tap"/>

                <TextBlock
                    Grid.RowSpan="3"
                    Grid.Column="0"
                    Margin="10,10,0,0"
                    x:Name="uniName"
                    Text="{Binding name}"
                    TextWrapping="Wrap"
                    VerticalAlignment="Top"
```

```
                    FontSize="25"
                    FontStyle="Italic"
                    Tap="uniName_Tap"/>

            </Grid>
        </DataTemplate>
    </phone:PhoneApplicationPage.Resources>

    <!--LayoutRoot is the root grid where all page content is placed-->
            <Grid x:Name="LayoutRoot">
        <Grid.Background>
            <ImageBrush ImageSource="/../images/background.jpg" Stretch="UniformToFill" />
        </Grid.Background>
                        <Grid.RowDefinitions>
                                    <RowDefinition Height="Auto"/>
                                    <RowDefinition Height="*"/>
                        </Grid.RowDefinitions>

                        <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <controls:Pivot Margin="8" Title="University of Portsmouth">
                                                <controls:PivotItem Header="info">
                    <Grid>
                        <ListBox>
                            <TextBlock
                                Text="The University of Portsmouth is a university in
Portsmouth, England. The University was ranked 60th out of 122th in The Sunday Times
University Guide. The University is a member of the University Alliance, a group of 23 major
business-focussed pre and post 1992 universities.&#x0a;&#x0a;The University is split between
two campuses: Guildhall and Langstone.&#x0a;&#x0a;Langstone is the smaller of the two
campuses, located in Milton on the eastern edge of Portsea Island, the island on which the
city of Portsmouth sits. The campus overlooks Langstone Harbour and it is home to the
University's sports grounds. It also includes a restaurant and bar, as well as a 'student
village', which provides accommodation for 565 students in three halls of residence ; Queen
Elizabeth Queen Mother (QEQM), Trust Hall and Langstone Flats. Students in QEQM and
Langstone Flats have en-suite rooms. It used to be home of the University's School of
Languages and Area Studies. The School has now moved into the Park Building on the Guildhall
Campus.&#x0a;&#x0a;The Guildhall site is much larger. Unlike most university campuses, it is
not all enclosed on one tract of land, instead featuring various university buildings
scattered throughout the centre of the city. This campus contains much of the University's
teaching facilities, and nearly all of the Student Halls of residence (except the Langstone
student village and two halls (Rees Hall and Burrell House) located on Southsea Terrace, the
city's main esplanade).&#x0a;&#x0a;The University Library (formerly the Frewen Library) was
extended in 2006 at a cost of £11 million. Originally due to open in October, ongoing delays
meant that it was not complete until January 2007, when it was opened by the crime writer P.
D. James. The University has also in recent years invested in the Faculty of Science, in
particular through the renovation of its aluminium-clad main building (St Michael's) which
is adjacent to the student halls, James Watson.&#x0a;&#x0a;A new faculty called
&#34;Creative and Cultural Industries&#34; was opened in September 2006. It aims to provide
a unique environment in which all aspects of creative thinking will flourish and develop by
combining creative schools from across the university.&#x0a;&#x0a;On 16 May 2007, Sheila
Hancock OBE was appointed Chancellor of the University. Ms Hancock is an actor and author
and received an honorary degree from the University in 2005 in particular recognition of her
services to drama."
                                TextWrapping="Wrap"
                                LineHeight="30"
                                TextAlignment="Left"
                                Margin="8"
                                />
                        </ListBox>
                    </Grid>
                </controls:PivotItem>
```

```
                <controls:PivotItem Header="history">
                    <Grid>
                        <ListBox>
                            <TextBlock
                                Text="The University was founded as the Portsmouth and
Gosport School of Science and the Arts in 1869. Due to the dependence on shipping and trade
to the city, the main function of the college was to train the engineers and skilled workmen
who went on to work at the city docks, as well as at the large Royal Navy dockyard situated
in Portsmouth. However, due to a decline in shipping and population since World War II, when
large swathes of the city were destroyed by German bombing, the college was forced to
diversify in terms of its syllabus and teaching in order to attract new
students.&#x0a;&#x0a;This steadily continued until the 1960s when, due to a massive
government-sponsored expansion in Higher Education, the college was renamed Portsmouth
Polytechnic. Along with this new name came the power for Portsmouth to award degrees,
accredited and validated by the centralised CNAA. The expansion of the polytechnic continued
and in the late 1980s, it was considered one of the largest and the best performing
polytechnics in the UK. Portsmouth was granted university status with the power to validate
its own degrees along with the other polytechnics in 1992, under the provision of the
Further and Higher Education Act 1992.&#x0a;&#x0a;The University of Portsmouth is managed in
accordance with Articles of Government approved by the Secretary of State. The Act also set
the general format for an Instrument of Government determining the membership, constitution
and organisational structure of Boards of Governors.&#x0a;&#x0a;The formal inauguration of
the University of Portsmouth was celebrated at a ceremony in the Portsmouth Guildhall on 7
July 1992."
                                TextWrapping="Wrap"
                                LineHeight="30"
                                TextAlignment="Left"
                                Margin="8"
                                />
                        </ListBox>
                    </Grid>
                </controls:PivotItem>
                <controls:PivotItem Header="buildings">
                    <ListBox
                        x:Name="BuildingsListBox"
                        ItemsSource="{Binding UniversityBuildings}"
                        Margin="12,0,12,0"
                        ItemTemplate="{StaticResource BuildingsListBoxItemTemplate}" />
                </controls:PivotItem>
                                    </controls:Pivot>
        </Grid>
    </Grid>
</phone:PhoneApplicationPage>
```

UniversityScreen.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using System.Data.Linq;
using System.Data.Linq.Mapping;
```

```
using Portsmouth_Guide.Model;

namespace Portsmouth_Guide
{
    public partial class University : PhoneApplicationPage
    {
        public University()
        {
            InitializeComponent();

            this.DataContext = App.ViewModel;
        }

        private void uniImg1_Tap(object sender, GestureEventArgs e)
        {
            if (BuildingsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = BuildingsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/LocationScreen.xaml?lat={0}&long={1}", dataOfPlace.latitude,
dataOfPlace.longitude),
                    UriKind.RelativeOrAbsolute));
            }
            BuildingsListBox.SelectedIndex = -1;

        }

        private void uniName_Tap(object sender, GestureEventArgs e)
        {
            if (BuildingsListBox.SelectedIndex != -1)
            {
                var dataOfPlace = BuildingsListBox.SelectedItem as Venues;
                NavigationService.Navigate(new
Uri(string.Format("/Screens/InfoScreen.xaml?name={0}&type={1}", dataOfPlace.name,
dataOfPlace.t),
                    UriKind.RelativeOrAbsolute));
            }
            BuildingsListBox.SelectedIndex = -1;
        }
    }
}
```